Automated Drone Calibration System

Final Design Review

Inspired Flight Calibration Team
calibrate@inspiredflight.com

Jackie Paik
Matthew Carlson
Zach Richter
Tyler Van Den Berg
Ryan Zhan

Mechanical Engineering Department
California Polytechnic State University, San Luis Obispo
4 June 2020

Statement of Disclaimer

Abstract

The final design review of the Inspired Flight Calibration Team senior project will detail the process used to complete a verification prototype of a drone calibration device and discuss lessons learned and suggestions for improving this device. Going from brainstorming and conceptual prototyping all the way through verification prototyping and testing, we were able to design a gyroscopic device that met Inspired Flight's needs for the flight sensor calibration of their drones. The mechanical design involved comprehensive CAD models and hands-on manufacturing. The mechatronics side of the project worked heavily with electrical wiring and writing custom software to communicate and run the calibration sequences. While we were not able to complete as much of this project as planned due to the COVID-19 pandemic, we delivered a working verification prototype along with our documented work to Inspired Flight so that it can be integrated into their calibration process.

Table of Contents

List of Tables

1.0 Introduction

This senior project was brought to California Polytechnic State University by Inspired Flight, a commercial drone manufacturer and distributer. Inspired Flight hopes to have a new calibration and test process for their drones. Currently, each drone is manually calibrated and run through a flight test before being shipped, which sometimes leads to inaccuracies in the flight controller system. The company is looking to automate this process to achieve more consistency as well as reduce the employee hours needed from their engineers to complete this task. Our team consists of five mechanical engineering students, Jackie Paik, Matthew Carlson, Ryan Zhan, Tyler Van Den Berg, and Zach Richter. Our interests and skills vary from robotics and mechatronics to mechanical design and manufacturing. This report will describe in detail the entire process from research and setting objectives that led us to a preliminary design, and how the analysis of our design created improvements and allowed us to select specific components and plan the manufacturing process and software structure. This design is elaborated on in detail within the Final Design, Manufacturing Plan, and Design Verification Plan sections. Potential risks and challenges are also examined, and a project timeline is presented in the Project Management section.

2.0 Background

The objectives were determined by interviewing the Inspired Flight engineers and conducting research. From this, we gained a deeper understanding of the problems with the current calibration process and learned from technical research on different sensors and their calibration methods. The information gathered from the research is described in the Background section of this report. It contains what we learned about the company's needs and wants as well as relevant information on topics that are important for understanding areas of technology and engineering that will be prevalent in this project.

2.1 Summary of the Meeting with Inspired Flight

During the first meeting with our sponsors from Inspired Flight, Marc Stollmeyer and Martin Bialy, they discussed the expectations they had for the scope of this project. This conversation serves as background for the research conducted. The project is to be twofold: the primary goal being to do the calibration of the flight systems and the secondary being to run the drone through a simulated flight test. The first goal would ideally be for us to build a system that can autonomously and accurately run a drone through a six-orientation flight controller calibration, a "level horizon" calibration, and an on-board compass calibration. An optional second goal would involve holding the drone in place while feeding preset signals to the motors and using sensors to verify proper drone operation (Stollmeyer). Due to lack of time and resources, we were unable to work on this optional second goal.

The current calibration is done by hand and is not very precise, having an exact process that can be autonomously carried out would save Inspired Flight time and money, and ensure that their drones can be shipped out more quickly. Currently 1 in 4 drones have internal sensor issue out of the box and 1 in 5 drones need recalibration after their first test flight. The current flight tests are all done manually, outside. If these could be replaced with an accurate simulated test-flight apparatus, this would save significant time and hassle for Inspired Flight engineers (Stollmeyer).

2.2 Current Process

In order to design a product to assist in Inspired Flight's drone calibration process, we learned about their current process. At the Inspired Flight facility, we observed a drone calibration and what each of the steps involved. An important component in the process is the calibration software, QGroundControl. This contains the entire user interface for flying and calibrating the drones. The software is entirely open-sourced, and we researched the user guide and calibration documentation for the program (Gagne). During the observation, we saw the different requirements for each type of sensor to be calibrated. The various steps, drone orientations, and possible errors for the calibration processes are detailed below. Errors stemming from inaccurate calibration have the potential to cause decreased stability or constant drift which negatively affect the performance of the drone. All these calibration motions, positions, and errors will be considered when choosing designs to prototype. The exact step-by-step calibration and possible errors due to poor calibration are found in Appendix A.

2.3 Drone Regulations

When researching about drone calibration regulations, we were surprised to find very few standards relating to the actual calibration process. As aircrafts, UAV regulations are primarily set by the FAA. What we did find was primarily about required drone licensing for various sizes and classifications, as well as pilot licenses and restricted flight areas. This includes a ban for flying unmanned aircrafts on the Cal Poly campus without Cal Poly UAS approval ("Unmanned Aircraft Systems"). The only information we found about preflight calibration was "Conduct a preflight inspection, to include specific aircraft and control station systems checks, to ensure the small Unmanned Aerial System (UAS) is in a condition for safe operation" (United States, Congress). This statement is vague considering the preflight calibration process varies based on the drone and flight controller in use. Due to the lack of specific calibration requirements, government regulations are not currently a significant factor.

2.4 Research Tables

During our research of technical sources, we found very little available information pertaining to automated drone calibration processes. Because of this, our research was split in to two major groups: robotic manipulators for other applications and details about relevant sensor technology. There is some information on calibration, but it's not specific to using technology to automatically calibrate drones. Table 1 shows the relevant patents to the project with a short summary of the patent and how it pertains to the project.

Table 1. Patent Research Summary

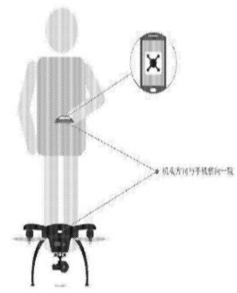
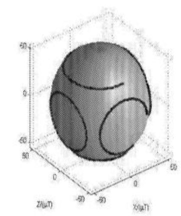
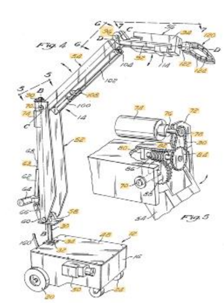
Patent	Summary	How this is helpful	Image
Yaw Angle Value Calibration Method and System for Unmanned Planes (CN107655470A)	This method utilizes a difference in value of the magnetic declination angle values, and the yaw angle value which is used for operation of the unmanned vehicle is compensated. The unmanned vehicle then regulates the nose orientation of the aircraft according to the compensated yaw angle value. (繆志豪)	This is directly related to calibration of UAV's; we can use this when doing tests to determine calibration accuracy	 图2 (繆志豪)
Ellipsoid fitting-based unmanned aerial vehicle magnetic sensor calibrating method and unmanned aerial vehicle (CN107894241A)	This might be very important when it comes to analyzing the magnitudes of the magnetic fields that effect our system. (智灵飞（北京）科技有限公司)	This might be very important when it comes to analyzing the magnitudes of the magnetic fields that effect our system	 图2 (智灵飞（北京）科技有限公司)
Unmanned aerial vehicle sensor calibration during flight (US10032275)	Research into doing/adjusting minute calibrations sensors while airborne. (Watson)	If we decide that we want a "smart" system that can correct itself and adapt, this is an interesting resource.	N/A
Mobile Robotic Arm (US5413454A)	"This invention relates to a mobile robotic arm which is adapted to grasp objects at low-level, intermediate level and high reach areas of a domestic dwelling." (Movsesian)	Even though we have not decided on an official design, a multi-jointed robotic arm is a very probable solution. This patent describes a basic robotic arm and has draws of servo motors and joint connections between sections.	 (Movsesian)

Table 2 describes the research results of similar products within the scope of the project. It can be seen in the table that not much exists for this particular process.

Table 2. Similar Product Research Summary

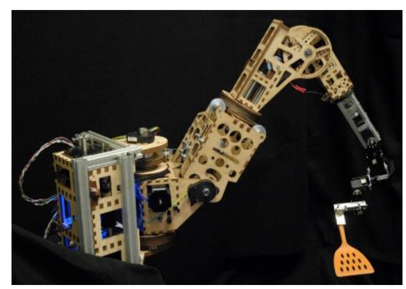
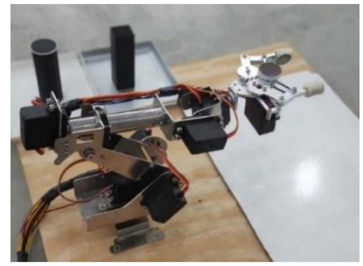
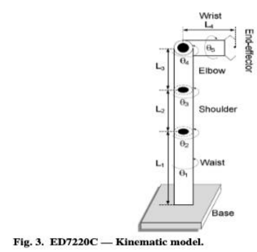
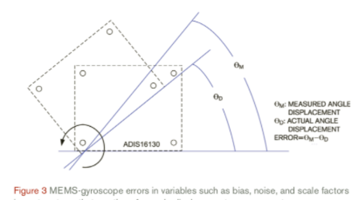
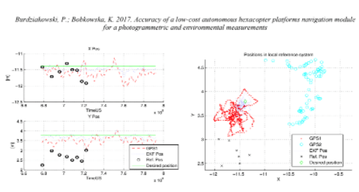
Product	Summary	How this is helpful	Image
Drone Calibration Turntable	Drone flying enthusiast created a simple turntable to more accurately and consistently calibrate his DJI Phantom quadcopter. This product will not be sold, so direct competition is not an issue. This is the only physical drone calibration we found with extensive research. (Mack)	This can influence our final design because a turntable is a very simple way to rotate a large object around an axis.	 (Mack)
7 DOF Robotic Manipulator	Students from Stanford University made a 7 degree of freedom robotic arm that has "significantly lower parts cost than comparable manipulators. It is made of laser cut plywood and can comfortably hold 4.4 pounds of payload." The whole machine was built for just over $4000. (Quigley)	Using a robotic arm is a viable option and due to cost and adaptability building one is a possibility. This article and product show that you can build a 7 DOF robotic arm for relatively cheap	 (Quigley)

Table 3 outlines the research conducted of journal articles relevant to the project and provides insight into said relevance.

Table 3. Journal Articles Research Summary

Journal Articles	Summary	How this is helpful	Image
Vision Guided Robotic Arm	Can visually identify an object and then grab it with a robotic arm. Uses force sensors to not over-grip. Relocates block to new location. Used premade 6 DOF robot arm, that uses 7 servo motors. (El Shair)	Using sensors as feedback for a robotic arm is impressive technology that we might need to use for accurate location and operation of a manipulator arm	 (El Shair)
Modeling and analysis of a 6 DOF robotic arm	This paper provides a mathematical model that, given a desired position and orientation of the robot end, will return the corresponding joint angles required to reach said position. (Iqbal)	This will be helpful in developing the control system if we decide to build a device such as the one talked about in the paper, especially when trying to set up the drone in the initial calibration position.	 (Iqbal)
A Simple Calibration for MEMs Gyroscope	These sensors are commonly referred to as microelectromechanical systems, MEMS for short. These sensors have error mainly due to bias, noise, and scale factor errors. (Looney)	It is important to understand the source of errors for the sensors in the flight control systems. The drones used by Inspired Flight utilize the PixHawk 4, which contain accelerometers/gyroscopes manufactured by Bosch and InvenSense.	 (Looney)
Accuracy of a Low Cost Autonomous Hexacopter Platforms Navigation Module for a Photogrammetric and Environmental Measurements	Used Ublox NEO M8N GPS with PixHawk flight controller. Compared positional error with the GPS directly on drone versus off 12 in on carbon fiber stand. Putting the GPS further improved positional error significantly (~5x). (Burdziakowski)	This is the same controller that Inspired Flight uses, and this is interesting and useful information to keep in mind when designing so that we don't cause unwanted calibration errors	 (Burdziakowski)

3.0 Objectives

The Objectives section will define what is included in the scope of this project. Outlining the major goals for this project is important for ensuring that the final design solves the problem that Inspired Flight has with their current calibration and test process. The scope also ensures that the tasks that our team has determined are accomplishable within the year long duration of this project. From the QFD we see that the most important quality of the device is the ability to reliably and accurately calibrate the drone. A Quality Function Deployment (QFD) is included as an attachment to demonstrate how the engineering specifications were chosen from the customer's needs and wants.

3.1 Problem Statement

Inspired Flight Technologies would like a way to calibrate then test the calibration of their unmanned aircrafts with a more consistent procedure, as the current processes are conducted manually; this can be slow, awkward, and a potential source of error.

3.2 Boundary Diagram

Figure 1 shows the scope of our work in a graphical manner. The system which we are to design is enclosed in the dotted line. Factors that will affect our design, but which are not in our control, are included on the boundary of our system. An example of this is the roof/metal objects that we will not change but will need to be aware of in order to avoid magnetic interference during calibration.

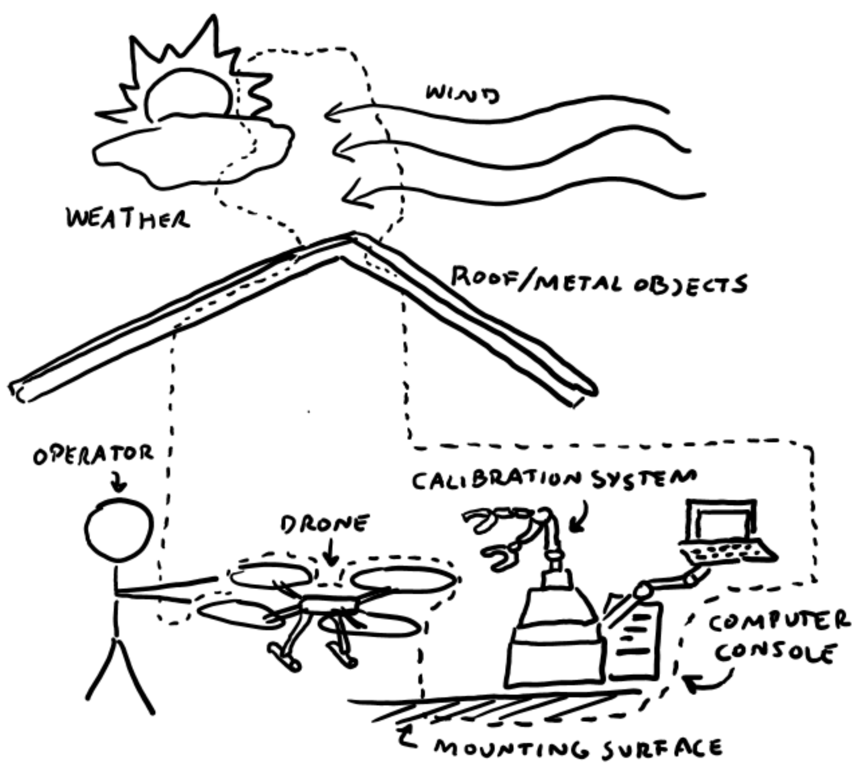


Figure 1. Boundary Diagram Demonstrating Our Scope

3.3 Specifications

To determine the specifications, we recorded the needs of Inspired Flight that the device must meet. These needs include the ability to fit through a doorway, be easily transportable, withstand the outdoors to some degree, be simple to use, and to be compatible with Inspired Flight's current and future drones (Stollmeyer). Other lower priority "wants" were also defined for the project. These needs were put into a Quality Function Deployment (QFD) and compared against more quantifiable specifications well as current alternatives to find the most important specifications. During the writing of the QFD, we ensured that all needs were accommodated for that there was no over-specification of the problem. Once completed, the priorities of the engineering specifications were determined. A completed QFD can be found in Appendix B and the entire list of specifications required of the device can be seen in Table 4. This table has been updated to reflect more relevant compliance ratings, as many of the simple measurements better characterized as inspections, since they do not require detailed test plans. The angle requirement was also adjusted from ± 0.1 degree to ± 1 degrees with the approval of the sponsor. This was done once we determined such a large high precision device would not be achievable with the given budget.

Table 4. Engineering Specification Table

Spec. #	Specification Description	Requirement or Target (units)	Tolerance	Risk	Compliance
1	Accurate drone calibration	±1 degree	Max	H	T
2	Number of set up steps	5	Max	M	I
3	Time to run	10 min	Max	M	I
4	Compatible drones	IF current and future drones	Y/N	M	I
5	Budget	$4,000	Max.	M	I
6	Weatherproof	Can be used outside	Y/N	M	I
7	Reliability	99%	Min	M	T
8	Size	4' x 4' base	Max	L	I
9	Lifetime	1,000 uses	Min	L	A
10	Weight	50 lbs.	Max	L	I
11	Fits through door	32" door frame	Y/N	L	I
12	Standard parts	When possible	Y/N	L	I
13	Uses wall power	120 V	+/- 5V	L	I

Risk is rated as high, medium, or low (H, M, L) and compliance is tested by test, analysis, or inspection (T, A, I).

1. Testing the accuracy of the drone calibration will be done by holding the drone at a specified angle and comparing the drone's measured orientation versus the drone's reported orientation.
2. A set of instructions will be drafted and sent to the sponsor for review to ensure that set up of the device can be clearly understood in 5 or less steps.
3. The machine will be timed using a stopwatch from start to finish.
4. In order to test compatibility with current and future drones, a current Inspired Flight drone will be attached to the device and run through the processes. Feedback from Inspired Flight will ensure future compatibility.

5. In order to measure the total cost of the final prototype, the prices of each component on the final device will be totaled. Because this is a prototype the amount of engineering hours will not be factored into the final cost.
6. Testing the ability of the device to survive outdoor conditions will not take place; however, it will be considered throughout the design processes (i.e. using materials that do not oxidize easily).
7. Reliability will be tested by running the final prototype 100 times and comparing the number of successful and unsuccessful runs.
8. The size of the device will be measured by placing it upright in a horizontal 4' x 4' area.
9. The lifetime of the device will be estimated by analyzing the stresses seen on the mechanical components as well as the through the provided datasheets of any electrical components.
10. The weight will be measured on a scale. This measurement will be compared to the total weight of each of the components for verification.
11. The team will physically move the device through a doorway at the Inspired Flight office to ensure that specification number 7 is met.
12. This will be a design consideration similar to item number 6.
13. The device will be powered by a standard wall outlet. This will be tested through observation.

3.4 High-risk Items

The highest risk specification is the accuracy of the drone sensors after calibration. This specific parameter was identified as a high-risk objective because of its importance to proper drone flight. Special consideration will be given to ensure the fulfillment of this specification. We are planning to have one or several alternative sensors to measure the orientation of the drone, then compare these measurements to the orientation reported by the drone itself. Due to complications with timing and the COVID 19 pandemic, not all specifications were tested. More information is provided in the Design Verification section.

4.0 Concept Design

The Concept Design section details the process to create a high-level design for this drone calibration automation project. This goes from brainstorming to deciding on a concept in a decision matrix. Also in this section are the methods and tools to help with concept generation and convergence. Note that this section is included to document the formulation of the design. For the most up to date design information, see Section 5.

4.1 Brainstorming

To obtain a well thought out design, we used multiple methods to encourage the creation of as many ideas as possible. To begin with, we used a slightly modified 6-3-5 brainwriting method. This method consists of each team member writing down ideas for two minutes, passing the list of ideas to the next person, adding to the new list, and repeating until everyone has seen every list. This method provided us our first ideas on how to meet our problem statement. Putting the ideas on paper helped visualize and provide more fuel for creativity.

The second method we used was the Stanford Brainstorming Activity. To start this off, a function decomposition was completed, as shown in Figure 2. With this, we determined the basic functions required of our design. Because some functions were repeated under multiple function groups, duplicates were combined. This produced the following list of functions: constrain movement, support weight, minimize vibrations, measure angle, apply force, measure position, and spin. For each function, we brainstormed as many ideas as we could, ranging from genius to absurd, allowing the creativity to flow. These ideas are all listed in Appendix C.

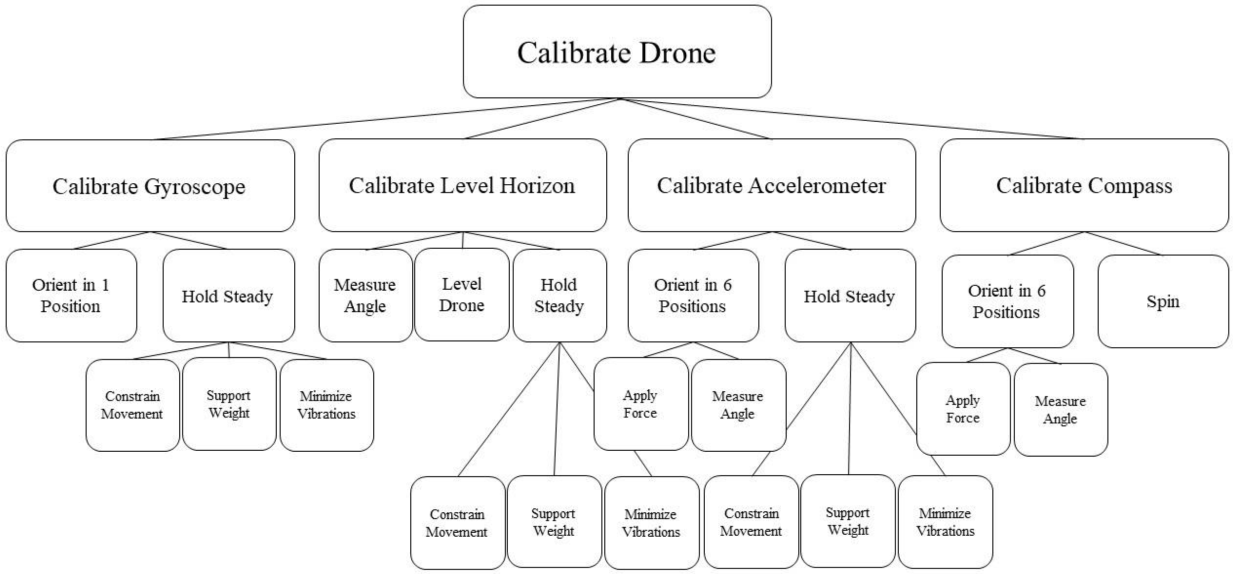


Figure 2. Functional Decomposition

4.2 Concept Models

With all the previously brainstormed ideas in mind, we moved to concept modeling. Concept modeling consisted of building models of the ideas that we had brainstormed for each function. These were all very basic models made from materials like pipe cleaners and foam board, simply for visualization, not function or accuracy. However, due to difficulty of modeling certain functions and interdependence of other functions, many concept models include multiple functions while some functions are never modeled. Figure 3 shows a collection of the concept models created. Most of the models here are demonstrating the means of attaching the drone, the overall structure, or the mechanism to rotate and orient the drone. Model 1a is developed further into the turntable arm and model 4 into the gyroscope, as seen in Table 6.

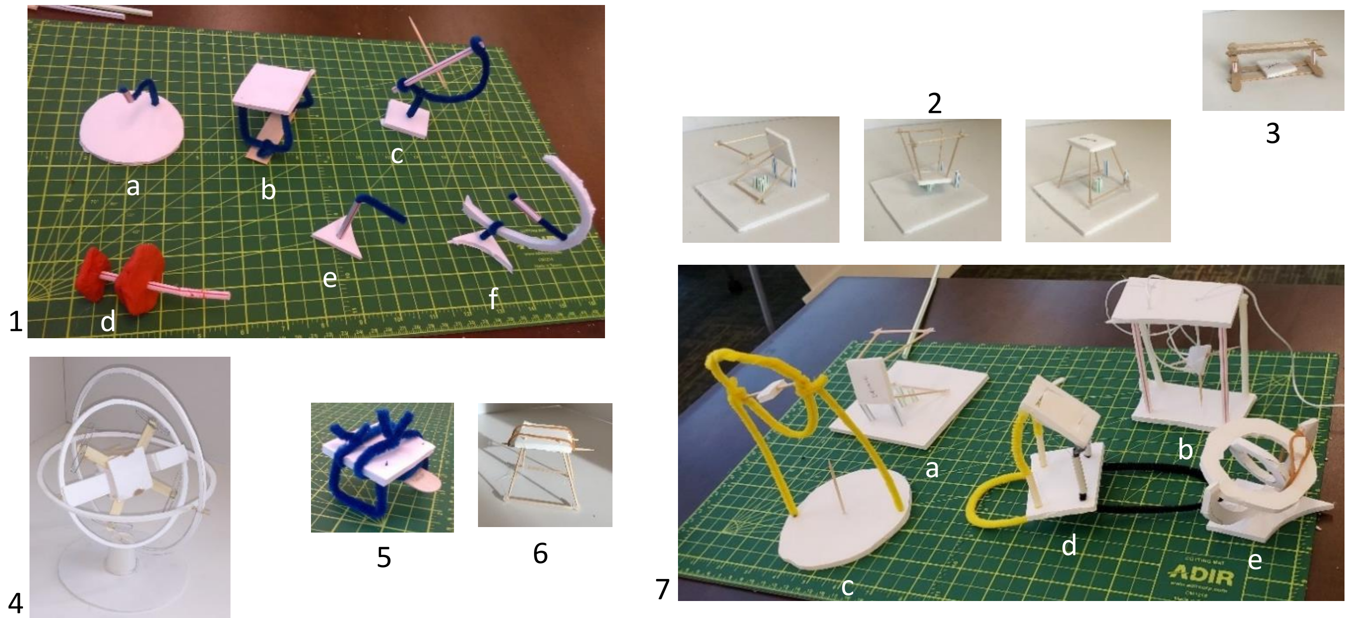


Figure 3. Concept models generated through ideation process

4.3 Decision Matrices

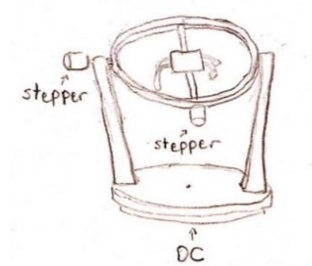
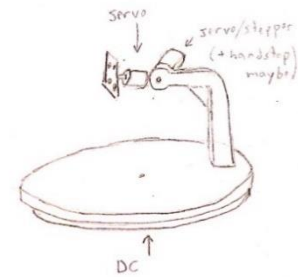
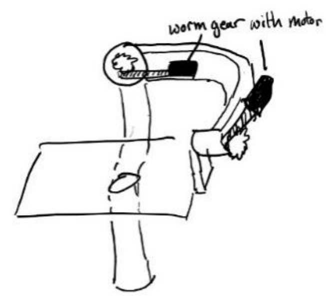
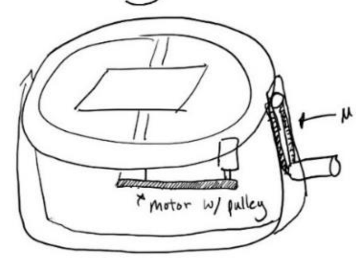
Following the creation of physical representations of the concepts, a series of Pugh matrices were developed and can be found in Appendix D. These compare several of the best concepts for a given function based on various requirements. These functions were revised from the earlier determination of functions. These functions are structure, attachment, drive motion, measurement, and drive train. The Pugh matrix results showed the concepts' abilities to meet the requirements. A morph chart, shown in Table 5, was created with the functional concepts and used to determine various combinations creating a single concept model.

Table 5. Morph Chart

Structure	Attachment	Actuation	Measurement	Drivetrain
Gimbal rings	Clamp sandwich	DC + encoder	Encoder	Direct
Camera gimbal	Bolt to base	Servos	IMU	Gearing
Turn table arm	Strap to body	Stepper motor	Potentiometer	Worm gears
Sliding-half moon	Strap to legs	Linear actuator	Bubble level	Belts
Cables	Square keylock		Stepper steps	Chains
			Hard stops	

The morph chart above shows all the functional concepts we developed, and an example combination is highlighted. These combinations were discussed and eventually we decided on the top four concept designs shown in Table 6. Each of the top concepts are described in more detail as well as a basic sketch.

Table 6. Top Concept Designs

Design Name	Description	Sketch
Gyroscope	This design is inspired by a gyroscope. It rests on a turntable. The ring rotates about a horizontal axis. The center bar also rotates about its own axis, giving it the third degree of freedom. On the central bar is a plate with straps. This will strap down the body of the drone.	
Turntable Arm	This design consists of an arm on a turntable. The arm has a joint which raises a linkage from straight up to straight down. The linkage has a motor a motor attached which rotates an adapter plate about the axis of the linkage. The adapter plate bolts on to the drone.	
Adapted Camera Gimbal	This design is an oversized camera gimbal on a turntable. The two degrees of freedom on the camera gimbal arm are driven by worm gears to move motors away from the drone and allow motion to lock while motors are off. The attachment point is a pair of plates which fit around the drone and clamp onto parallel flat surfaces of the drone.	
Pulley Driven Rings	This design uses 3 gyroscopic rings powered by motors. The base axis of rotation is driven by a belt pulley, allowing the driving motor to be moved away from the drone. The drone is attached to a plate on the axis of the central ring by straps.	

These first four concepts were put into a weighted decision matrix in Table 7. This compared each design to a datum design in terms of the requirements specified during the problem definition portion of this report. To determine the score for each design, certain criteria required basic engineering analysis while others were scored qualitatively. In this case, the design datum is the gyroscope design because it was the most likely design to proceed with. After the scores were multiplied by the weighted importance of the criteria, each design had its total score summed. Two designs came out clearly above the others: the gyroscope and the turntable arm. Our team decided to build prototypes of these top two designs to aid in the final decision.

Table 7. Decision Matrix

Criteria	Weight	Gyroscope	Turntable Arm	Adapted Camera Gimbal	Pulley Driven Rings
Low Cost	2	0	1	0	-1
Compact	2	0	1	1	-1
Robust	3	0	-1	-1	-1
Manufacturable	4	0	1	1	-1
Vibrations	5	0	-1	-1	0
Survive Elements	2	0	0	0	0
Weight	3	0	1	1	-1
Power Consumption	1	0	-1	-1	-1
Accuracy	4	0	0	0	1
Reliability	4	0	-1	-1	1
Total		0	-2	-4	-7

4.4 The Decision and What Follows

The purpose of these preliminary designs is to obtain feedback and receive confirmation from Inspired Flight to move forward with a single design. The two competing designs are the gyroscope and the turntable arm. Concept prototypes were made of each design in order to gain a clearer understanding of the pros and cons of each. These can be seen in Figure 4, Figure 5, and Figure 6. After further consideration and prototyping both designs, the gyroscope design was chosen. A high-level CAD model of the chosen design was then created in SolidWorks.

4.4.1 Mechanical Design

The gyroscope shown in Figure 4 holds the drone in the center plate. It offers 3 degrees of freedom (DOF) and supports each component with two points of contact. We plan to drive each joint with a motor. In this design, the center of the drone rests on the axis of rotation for all yaw, pitch, and roll. We also see that the motor on the middle ring may be difficult to power and that clearances in all orientations must be considered.

Figure 4. Concept prototype for the gyroscopic design, laser cut from MDF

The turn table arm shown in Figure 5 holds the drone on the square plate. The entire structure rotates on a turntable. An arm extends up from the table. At the end of the arm is a motorized joint to a linkage, which can be angled upwards, straight out, and downwards from the supporting arm. A secondary motor rotates the adapter plate at the end of the linkage. This rotation is axial to the linkage, giving us the final DOF of the 3 DOF that is required for the calibration process. One concern is that the center of the drone will shift off the axis of rotation as the device changes orientations. Also, integrating the secondary motor on the structure of the linkage could be difficult, especially when it comes to supporting weight at the end of the linkage.

Figure 5. Concept prototype for turntable arm, laser cut from MDF

From the physical prototypes, the nuances of each design become clear. The gyroscopic design has some key benefits. The gyroscope inherently positions the drone's center of gravity over the axis of rotation, where the turntable arm cannot do so in all orientations. Both designs will allow the necessary degrees of freedom to orient the drone for calibration; however, the turntable arm is expected to experience a much greater moment on the arm of the device, possibly increasing the amount of power required to run the motors and inducing stability issues. The turntable arm also requires actuators to be much closer to the drone itself, which may cause magnetic interference. The turn table has the benefit of positioning the actuators on one side of rotation, meaning that the wiring required for the motors will not become entangled with the rest of the device. The turn table also has a more structurally simple design and may require less complex manufacturing.

We plan to move forward with the gyroscopic design due to the aforementioned benefits. A preliminary CAD model of this can be seen in Figure 6. We believe that the benefits that the turntable arm offer do not outweigh the strengths of the gyroscopic design. This design will allow a more stable orientation processes and offers more support during calibration. To manufacture the curved geometries required for the gyroscope the team plans to use the waterjets available on campus. For smaller, lower risk components of the device, the team has access to various 3D printers across the Cal Poly campus and at Inspired Flight.

Many of the components labeled in the isometric view represent the basic structure and layout of the concept design. Few specific components, such as the motors we will be using, have been decided on. The material has not been selected, but we plan on using a single thickness of sheet material for all the structural parts for ease of manufacturing. The bearings to support loads at the joints while still allowing for rotation have not yet been selected or sized based on the loading conditions. Gussets provide rigidity to the base of the structure. The aircraft is placed in the center of all axis of rotation for a more balanced device. Planed analysis and testing for further specifying the components are discussed in Section 5.3 of this report.

Figure 6. Isometric concept CAD model of gyroscope device

4.4.2 Electronic Hardware Design

The calibration device will require several electronic hardware components for full autonomous functionality. Of these, electronic actuators are of particular importance. We are currently considering stepper motors, servo motors, DC motors with encoders, or some combination of the three for our design. This design decision will be finalized as the mechanical design is further analyzed, and the torque and accuracy requirements become clear.

The controller for the device will require serial communication in order to properly interface with the QGroundControl software running on the computer as well as the ability to send the appropriate control signals for the decided motors. For prototyping purposes, the team is planning to use an Arduino Uno with a motor driver shield due to its to adaptability and ease of use as well as its wide range of functions. In order to measure the orientation of the system for feedback and control, the device will utilize the measurement system of the actuators with the possible addition of an IMU for more verification of the orientation.

4.4.3 High Level Software Design

Currently, we expect the control of the calibration device to be separated by several tasks that will run concurrently, each directly controlling a specific function of the device. There will be a task for each motor, a task for measuring the current physical state of the system, and a task for interfacing with QGroundControl. Ensuring that the tasks are working cooperatively with one another will be handled by a Master Mind task.

Table 8. List of expected program tasks and their functions

Task	Name	Function
1	Master Mind	Ensure cooperative function between all tasks.
2	Motor Control	Control operation of the motors
3	QGroundControl Interface	Communicate with QGroundControl in and update the calibration device on the current state of calibration
4	Measurement	Measure the current state and physical orientation of the device

An initial state diagram of the Master Mind task can be seen in Figure 7. This Figure also shows the general process that the device will use to calibrate the drone. The device will first initialize and then wait for a signal from QGroundControl that signifies that the calibration processes has been started. Once this signal is received, Master Mind will signal the motors to begin operation so that the drone is properly oriented for a specific calibration. The measurement task will report the current orientation of the system and the Master Mind task will automatically begin the calibration processes. Once the correct orientation has been reached, either the base rotation will occur, or the drone will be held still. Once each calibration step has been completed, the calibration device will reorient and repeat until the entire calibration process is complete.

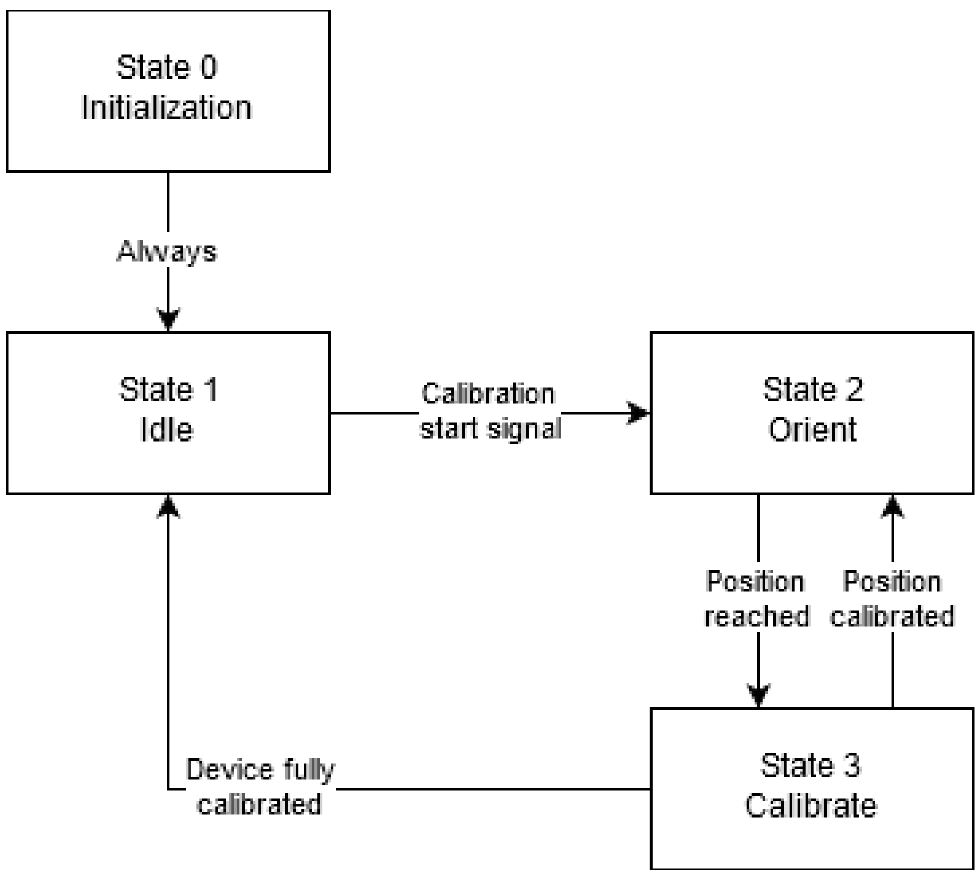


Figure 7. Diagram of General Organization of Calibration

4.5 Preliminary Analysis

The basic analysis of the top two concept prototypes shown was based on a single factor that led to the decision to pursue the gyroscope design. The gyroscope has more balanced force flow through the structure, which is advantageous as it evenly distributes loads and decreases moments at the motors.

The analysis for the turntable arm was modeled like a cantilever beam, which would produce much higher moments at the arms joint. This could also lead to larger deflections and increased susceptibility to vibration. These observations along with the overall user interaction with the prototypes swayed our decision towards the gyroscope design. The preliminary calculations are shown in Appendix E. The calculations show that the torque needed to rotate the balanced gyroscope design is within the range of commercially available stepper/servo motors.

4.6 Design Hazards and Unknowns

As part of our initial design proceses, a design hazard checklist was completed to identify any potential dangers our device may pose. The completed checklist can be found in Appendix F. The rotation of the device is a potential source of injury because of the pinch points it may create. To minimize this risk, the final design will include an emergency stop that will cease all operations of the device as well as labels identifying the danger this device imposes.

As previously discussed, challenges we expect to face with our design are handling magnetic interference and driving the joints of the device. We are unsure how much of an effect the magnetic fields generated by the electronic actuators of the device will have on the sensors of the uncalibrated drone and would like to perform additional testing. The unusual orientations required of the device for the calibration process may also pose a challenge when trying to attach actuators. Mounting motors in a compact manner without interfering with the rest of the device may be difficult and while there are devices (i.e. slip rings) that allow electric signals to be passed through rotational joints, further ideation is required for this aspect of the design. More detailed analysis must be done on the physical design to define the torque requirements (both dynamic and static) of the motors as well as the required strength of materials that will comprise the device.

5.0 Final Design

The Final Design section of this report details the design choices we made when further specifying the preliminary design as well as all changes made to the design during manufacture and assembly. Images of the entire design as well as more detailed views of important subsystems are included. Also included are samples of the calculations done to ensure that the design will meet each of the engineering specifications that were determined.

5.1 Mechanical Design Model and Diagrams

This section outlines the design of the physical structure of the drone calibrator. First, it discusses the intended design, and then it notes the changes made in the final construction of it.

5.1.1 Original Structural Design Goals

Images of CAD models showing the complete product, as well as more detailed images of smaller subsections and components are contained in this section. It is meant to provide a visual representation for our plan for the verification prototype. Additionally, Appendix G contains an exploded view of the structure.

Figure 8 shows a CAD image of all subsystems of the final design. A human has been added to show the size scale of the device. The four main subsystems have been labeled and will be shown individually in more detail in Figures 8-12. The three axes of rotations have also been in Figure 8. The primary axis of rotation rotates the outer fork, inner ring and drone mount system. The secondary axis of rotation, which lies through top of the fork system, rotates the inner ring and drone mount system. The tertiary axis of rotation acts through the inner ring and rotates only the drone mount system. The combination of these rotations nested within one another, allow for the drone to be placed in any orientation.

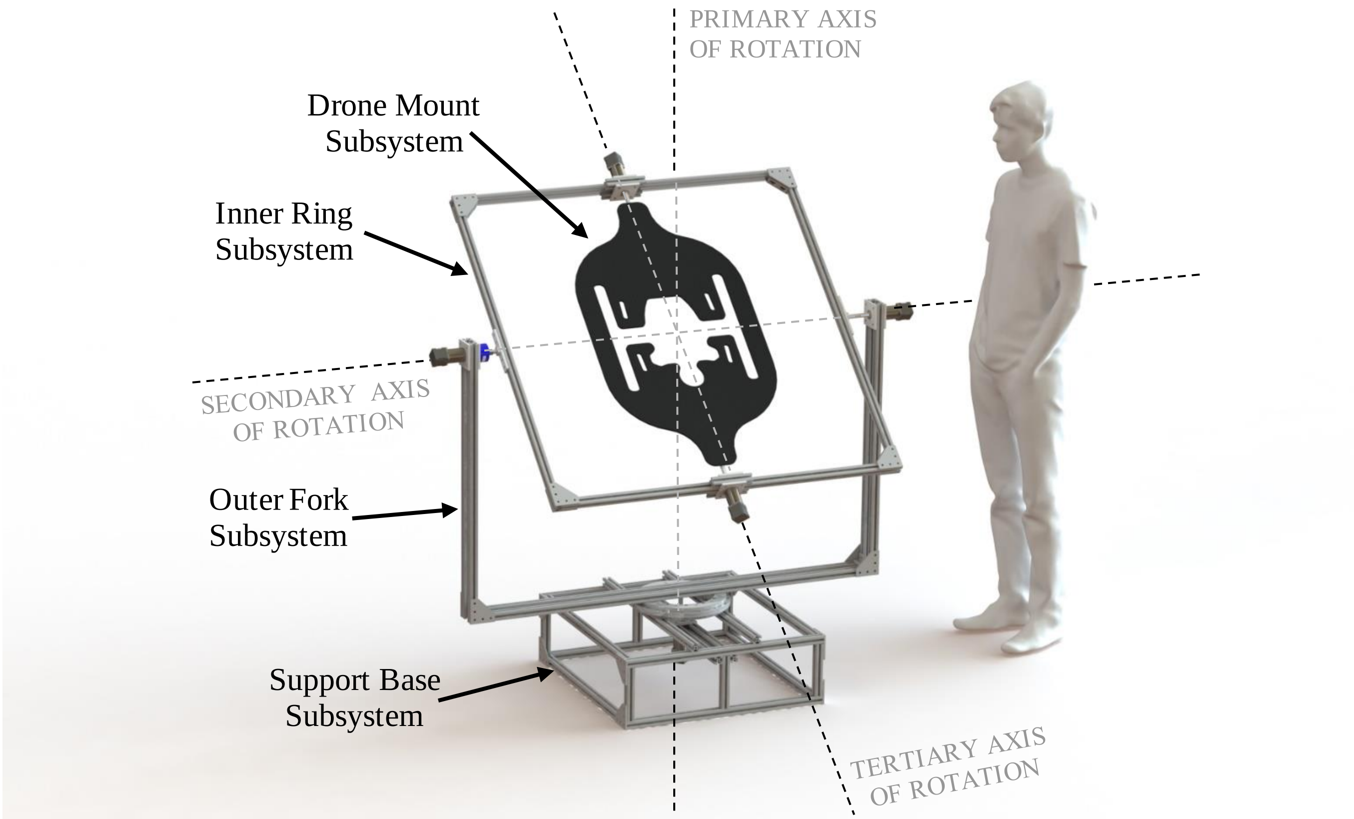


Figure 8. Mockup of Structural Prototype Design

Figure 9 shows the extent of what we decided to make for the structural prototype. The prototype includes two main subassemblies, the inner ring and outer fork, which cover a broad range of items to test for the final assembly. The useful concepts that this prototype will be proving, and testing will include controlling two motors simultaneously, spinning a large part of the frame, and passing signal and power wires through a slip ring.

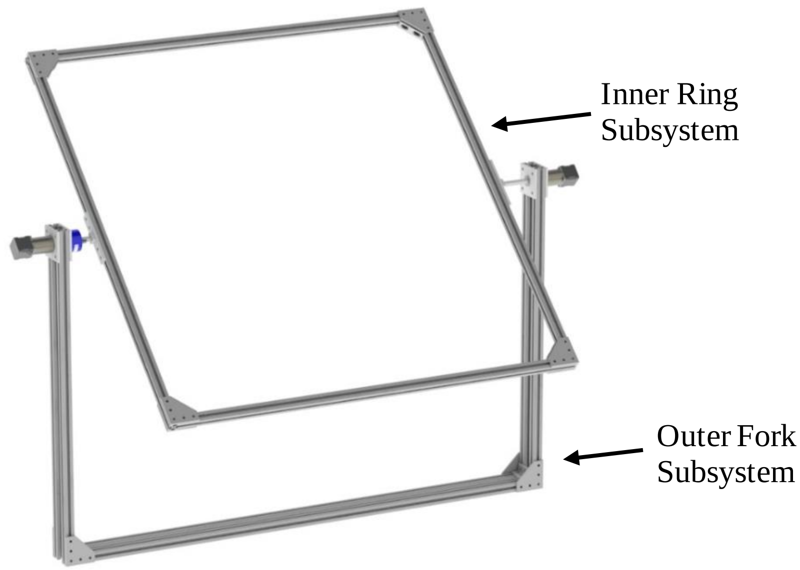


Figure 9. Overall View of Structural Prototype

Figure 10 shows the inner ring for the structural prototype which contains four equal lengths of 8020 fastened together to form a square. The ring is rotated by the hex shaft which is mounted to a connector plate fastened to the outside of the 8020.

Figure 10. Inner Ring Subsystem on Structural Prototype

The fork subsystem is shown with motors attached at the ugly joints in Figure 11. This fork system has been assembled as part of our structural prototype. Once the components have been tested, this fork will be integrated into our final prototype.

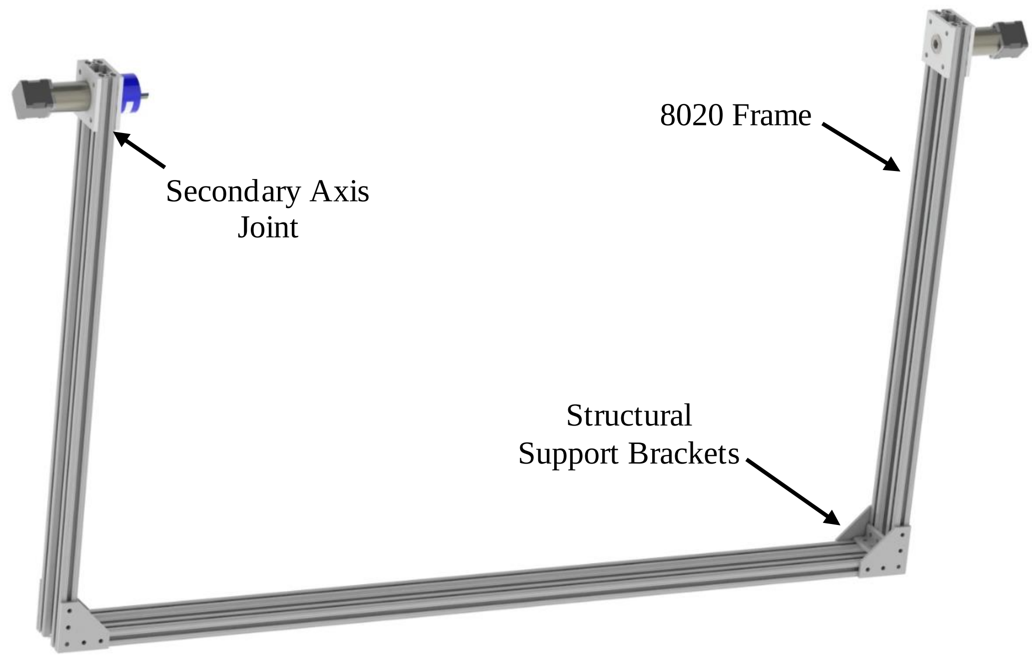


Figure 11. Outer Fork Subsystem on Structural Prototype

Figure 12 shows many of the critical components of the design as it is how the rotation about the axis is generated. The three mounting plates are all custom manufactured, but all other components were purchased. The stepper motor is geared down with a high precision gearbox to create adequate torque as well as decrease the speed. The slip ring allows for electrical circuits to be passed through the rotating joint, allowing for the motors on the tertiary axis to be powered. The motors drive a hex shaft, so the attachment to the ring is more secure and can transmit more torque.

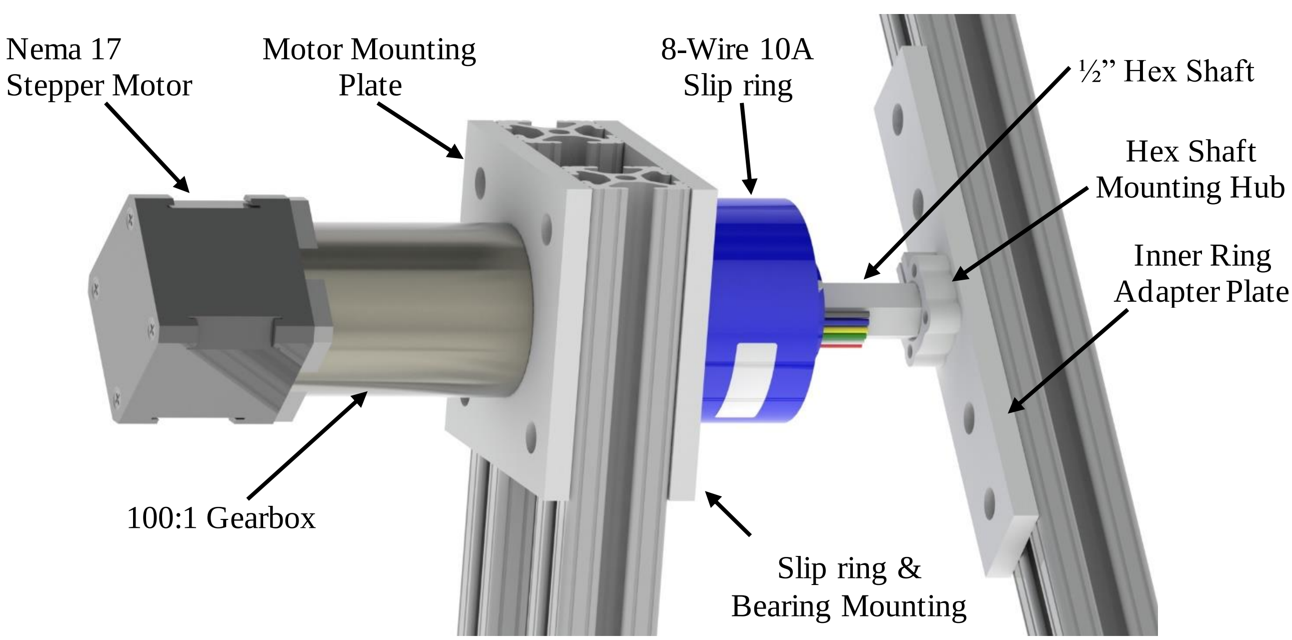


Figure 12. Detail View of the Secondary Axis Joint

5.1.2 Final Structural Result

During Winter Quarter of the project, building the prototype taught us quite a bit about the functionality of the system and led us to make a few major alterations to the design. We will now discuss these alterations that make up the final design of the physical calibration system. A rendering of the final design, labeled to show these changes can be seen below in Figure 13.

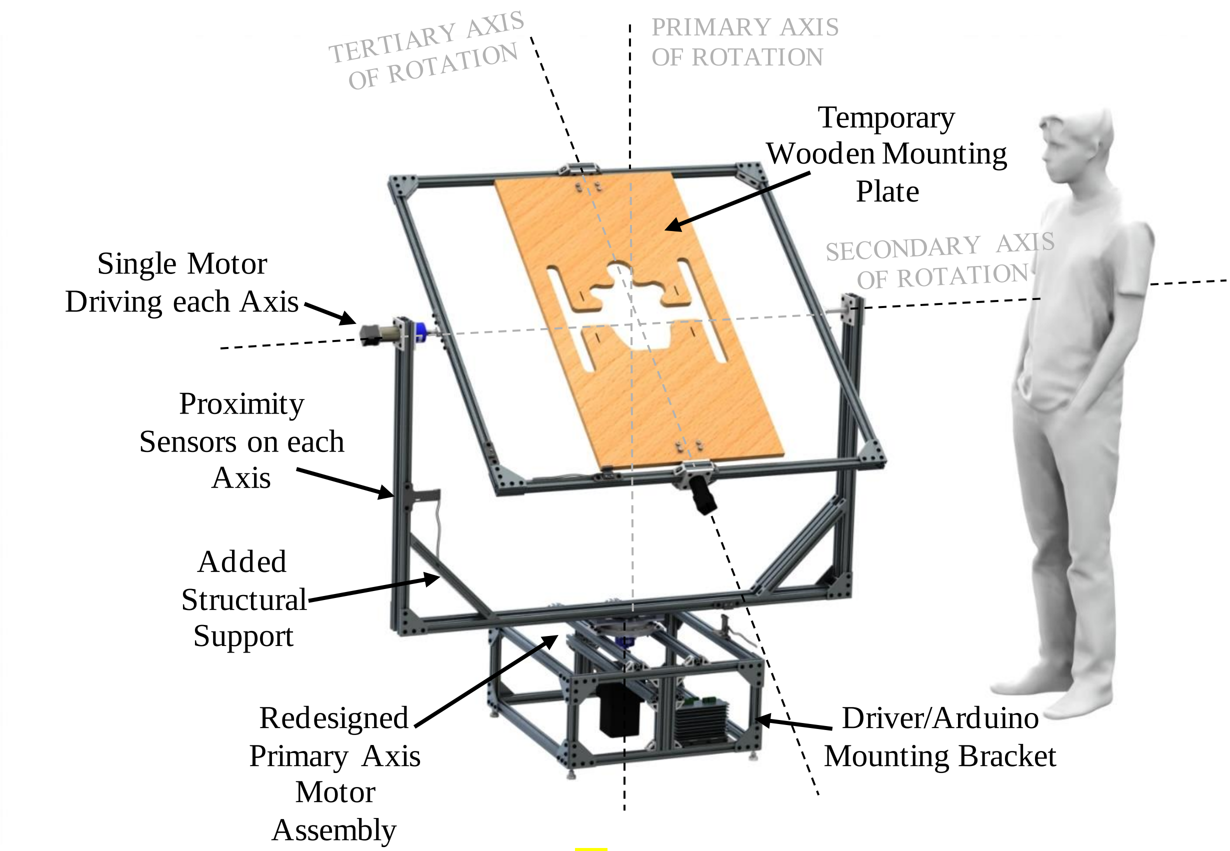


Figure 13. Rendering of Final Design

First and foremost, we decided to move to having one motor driving each axis instead of two. We were initially concerned that the motors would not provide enough torque alone. However, through testing, we determined that the torque was sufficient. Additionally, with two motors, wiring would be an issue because driving two motors with one driver proved to be ineffective.

Next, we decided to cut the drone mounting plate from plywood to test with, and with the COVID 19 pandemic, we could no longer manufacture the ½” polycarbonate sheet we purchased. Additionally, we decided that since the drone we wanted to mount was rapidly changing at Inspired Flight, it did not make sense to cut up a $300+ sheet of polycarbonate that could be useless on the finalized version of the drone anyways. We were able to use plywood to make a drone mounting plate perfect for the aircraft we were given to test with. The manufacturing section (6.2) of this report contains instructions to cut the mounting plate from polycarbonate, as that would be a better solution for a finalized product.

After constructing the prototype, we also decided to add structural support to the fork. The triangular 45° supports we added gave the entire frame more stability, especially while accelerating.
Next, the electronic hardware, namely the three motor drivers and the Arduino (discussed in the next section) needed a permanent home. We designed and 3D-printed a mounting bracket for the motor

drivers and Arduino that mounts in one of the inside corners of the base. Because the mount was essentially a cantilevered beam, we decided to print the bracket using polycarbonate filament, ensuring strength and rigidity. A close-up rendering of this assembly can be seen in Figure 14.

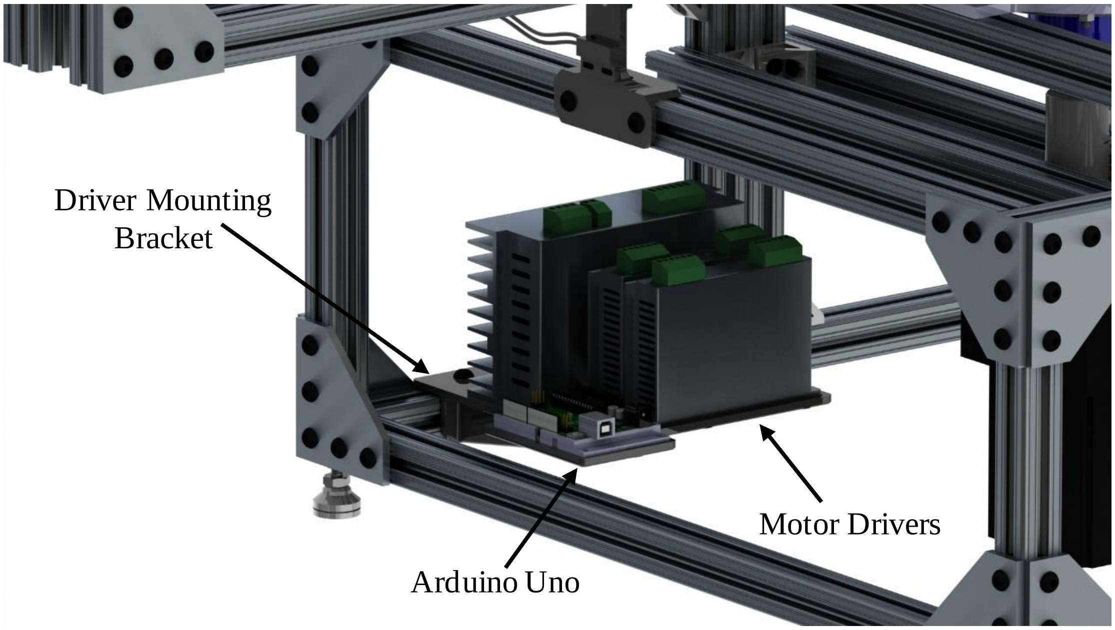


Figure 14. Close-up of Driver Mount Assembly

The next realization we had was that the system needed feedback for each axis' orientation. We determined the best way to accomplish this was by using proximity sensors that send an electrical signal to the computer when close to a magnet. Using the same polycarbonate filament, we designed and printed brackets for the proximity sensors, ensuring that each sensor was the proper distance from its corresponding magnet. Each sensor and magnet has a mounting bracket attaching to the 8020 frame, apart from the tertiary axis magnet, which was screwed into the corner of the drone mounting plate. Two of the three axis' sensors/magnets can be seen in Figure 15.

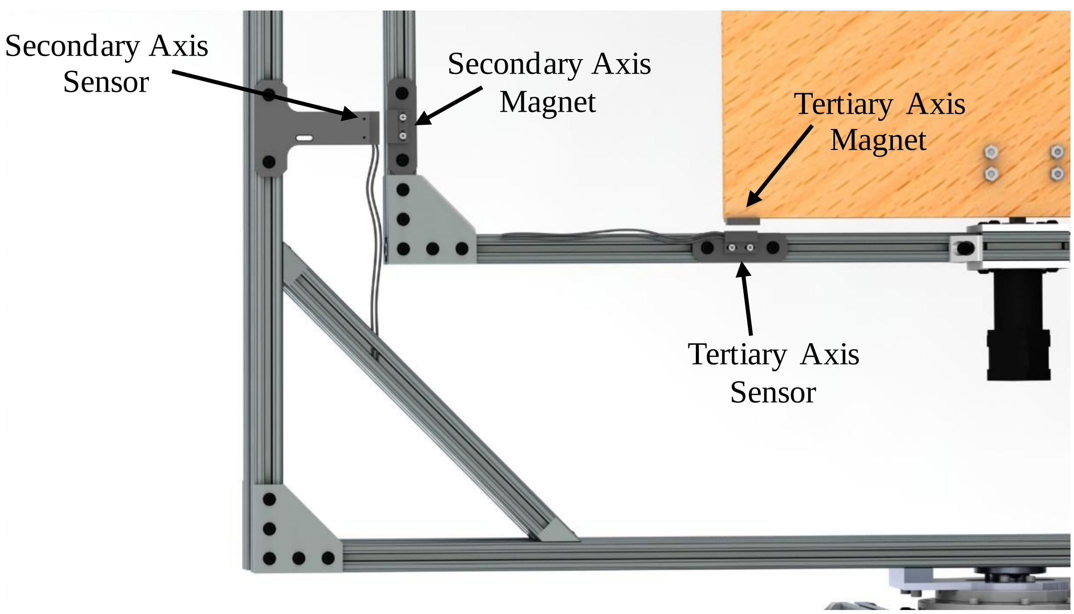


Figure 14. Close-up of Proximity Sensors

Lastly, the biggest change to the final design was the primary axis motor assembly. When researching the best way to transfer torque from the primary motor to the 8020 of the fork, we quickly realized that there are no standard or existing assemblies that make this possible. We had to be creative with the torque conversion and ended up using multiple diameters of parts, tying them together with a custom turned shaft. A section view of these assembled parts can be seen in Figure 16. In summary, the primary motor is mounted to the underside of the base via a custom 3D-printed polycarbonate mounting plate. The motor's 15mm shaft is then inserted into a 15mm coupling. The custom keyed shaft is has a diameter of 15mm at one end, which is inserted into the opposite side of the 15mm coupling. The custom shaft must then travel through a slip ring, which has an inside diameter of 12.7mm. To make this possible, most of the shaft is turned from 15mm to 12.7mm. To join properly with a 14mm flanged shaft mount (attached to the 8020 of the fork), a brass sleeve adaptor bushing(108) is placed on the end of the shaft to bring the diameter back to 14mm. The brass sleeve (108) (ID=12mm, OD=14mm) must be drilled out to 12.7mm on the inside to fit over the shaft, and a 5mm slot must be milled along the length of the sleeve to provide clearance for the key on the shaft. There will then be two of these keys (5mm x 5mm) that are pressed into both sides of the shaft, allowing torque to be transferred. Although the torque is transferred completely through this shaft, the shaft bears none of the fork's weight. To eliminate rotational friction while bearing the load of the fork, a sleeved turntable was placed between the base and the fork and is attached independently to both. This essentially allows the fork to be supported by the turntable and turned by the shaft assembly.

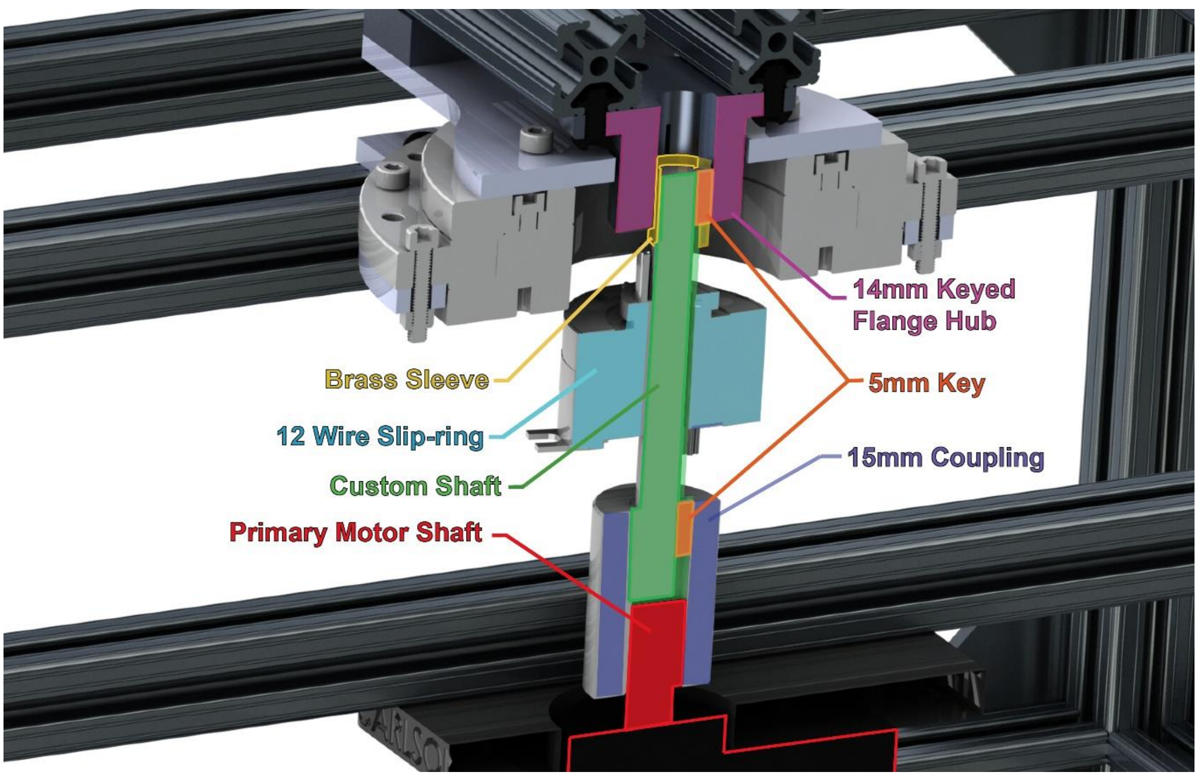


Figure 15. Primary Motor Torque Conversion

5.2 Final Electrical Design

In this section, the final electrical design is discussed. We will compare aspects of the original design to what we were able to implement in the final prototype.

5.2.1 Original Electrical Design Goals

The electrical design will consist of three motors and the supporting electrical hardware. Each axis will be driven by its own individual motor. The wiring will be routed along the appropriate beams and through the base and fork joints using slip rings. Motors on the same axis will be driven by the same signal. It has yet to be determined if this signal can be from one motor driver or two motor drivers receiving the same signal. This will determine the need for either three or five motor drivers. The signals to the drivers will be sent by an Arduino Uno and which will be connected by USB to a control computer. The wiring diagram is included in Appendix H. Regarding power, the Arduino will be powered by the computer and the motors will be powered by an exterior power supply through the motor drivers.

5.2.2 Final Electrical Result

After testing the structural prototype, it was determined that only a single motor was required for each axis. A NEMA 17 stepper motor with a 100:1 precision gearbox drives the tertiary and secondary axis. A NEMA 34 stepper motor with a 20:1 precision gearbox is used to drive the primary axis. One motor driver per motor is used, as per our original design. An emergency stop was also added to the system for a quick shutdown of all motors. When activated, the emergency stop shorts the MF- pins on the motor drivers to ground, deactivating the motor drivers.

5.3 Final Software Design

The software which we designed is in two parts. The first is written in Python. This will run on the controlling computer and will interface with the PixHawk (the drone module being calibrated), run the Master Mind code, and communicate over serial with the Arduino Uno. The second portion of software to design will be in C++ on the Arduino Uno. This code will interpret motor commands from the controlling computer and send them to the motor drivers. Figure 17 shows a communication diagram between the drone, computer, and Arduino.

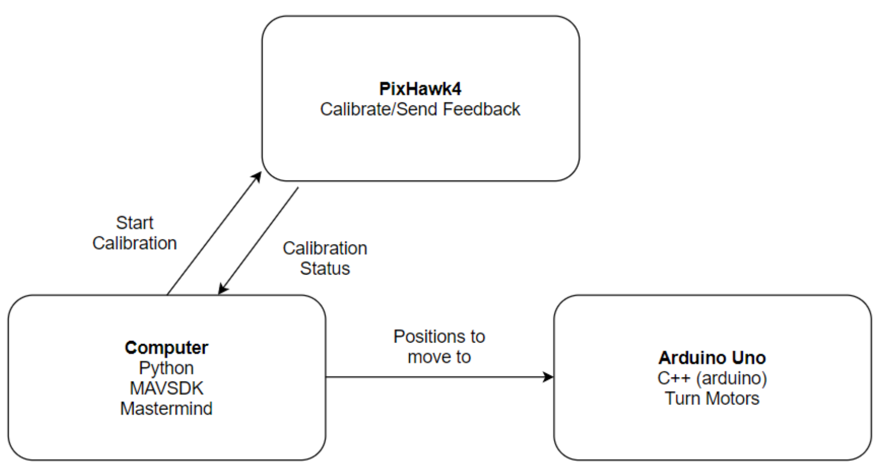


Figure 16. Communication Diagram of Three Main Computers

5.3.1 Original Arduino Software Design Goals

The Arduino code is broken into states. The states are shown below in Figure 18. In the first state, each of the three axes is run until it detects a reed switch, then it rotates to the starting position for calibration. In state 1, it waits for a command from the computer running the Master Mind python code. Positive one will spin the primary axis for about seven seconds. Positive and negative two will spin the secondary axis in either direction 90°, same for positive and negative 3 for the tertiary axis. To turn the motors on and off, the commands are 8 and 0 respectively. For each of the spin commands, (-3, -2, -1, 1, 2, 3), the code advances to state two ramp up to speed, state three to turn the appropriate amount, and then state four to ramp to a stop, returning to state one to await another command. Upon entering each state, the Arduino will send its current state to the Master Mind python code for verification.

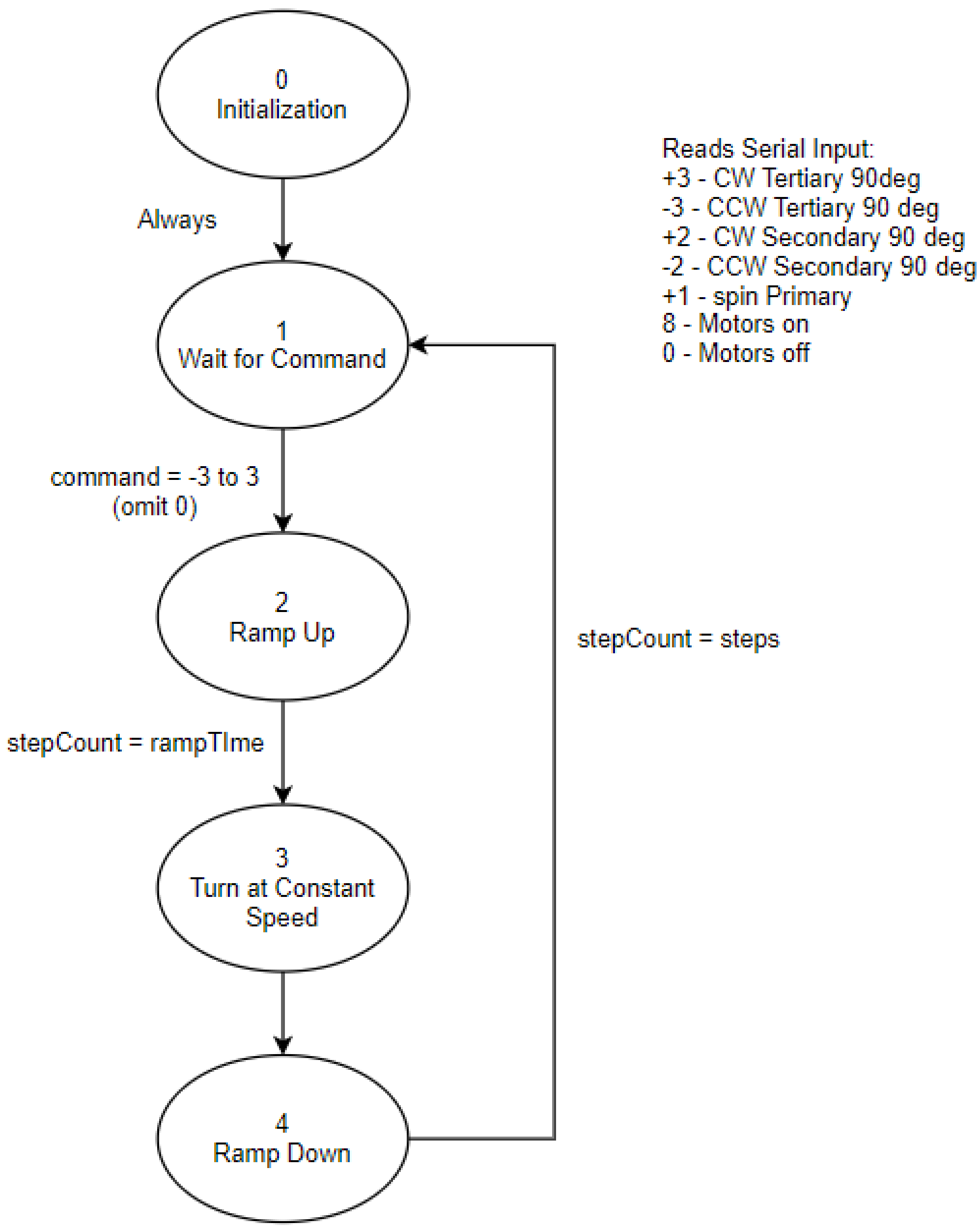


Figure 17. Original Arduino State Diagram

5.3.2 Arduino Software Results

Our Arduino code functions largely the same as originally designed. An additional software state was added for more functionality, which can be seen in Figure 19. This state, which is state 5 in the code, runs the motors one at a time until the reed switch on each axis is triggered. Once triggered, the motor will orient the axis to its starting position. This orientation state can be accessed by sending the Arduino a spin command of 9. This allows for the machine to be re-oriented at any time. A change to the overall structure of the organization of the Arduino code was also made. Originally, the Arduino only checked for a command while state 1. This functionality was changed to happen every cycle of the Arduino. This allows for a 0, or stop command, to be received and interpreted at any time. All other commands are only interpreted in state 1. The complete documented Arduino code can be found in Appendix I.

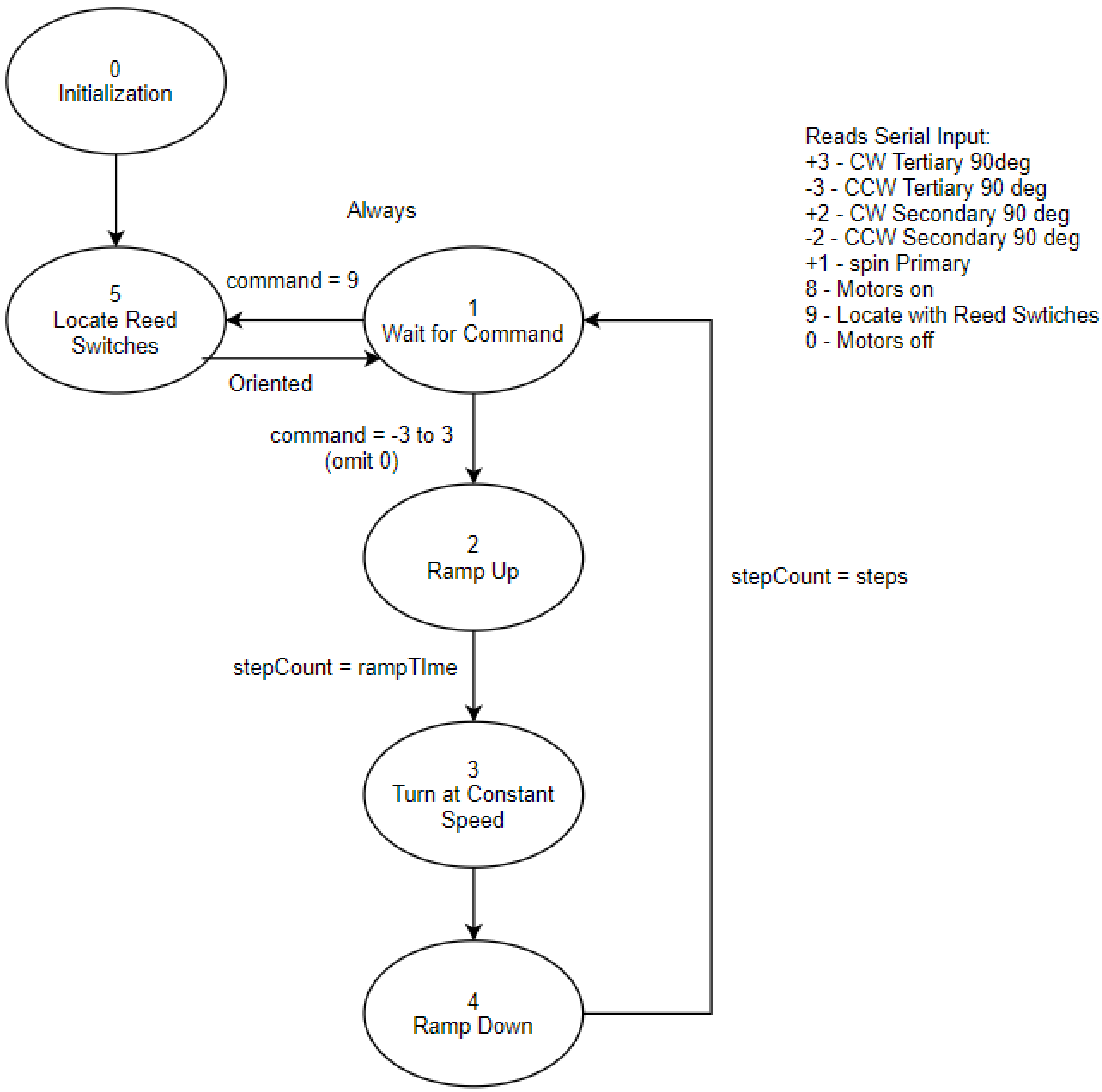


Figure 18. Final Arduino State Diagram

5.3.3 Original Master Mind Design Goals

Figure 9 shows the state diagram of Master Mind. Master Mind must calibrate four sensors. For each calibration, a similar path, focusing on the motors, is displayed below. The "Orient" state will signal the motors to move the drone to each of six orientations. The "Hold" state will be a delay for the PixHawk Module to calibrate. The "Rotate" state will rotate the base while the PixHawk calibrates. Once each sensor is calibrated, Master Mind will move on to the next sensor calibration.

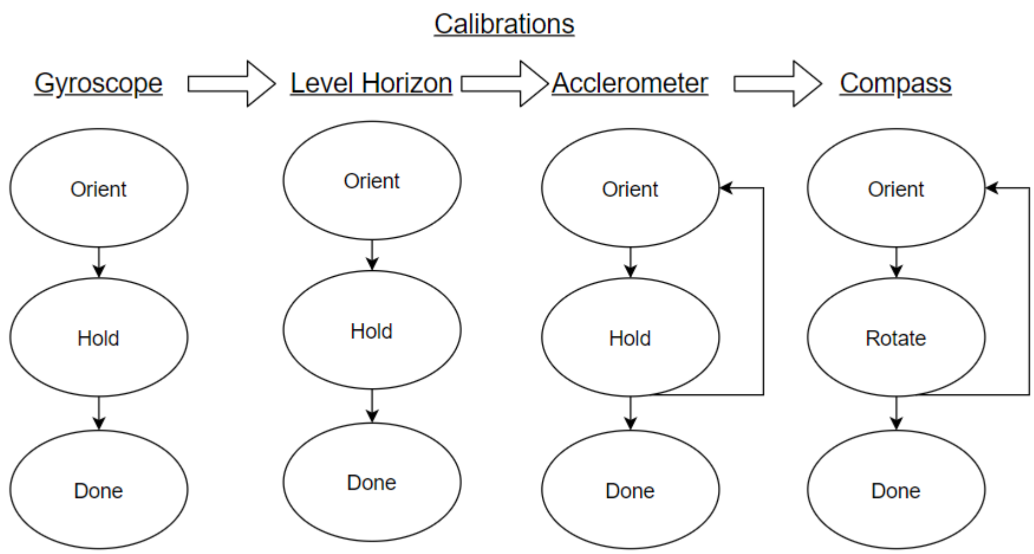


Figure 19. Original Master Mind State Diagram

5.3.4 Master Mind Results and Organization

The final python code consists of three tasks and a application to run the guided user interface (GUI). The GUI is shown in Figure 21. The guided user interface is created using Tkinter. One task handles the communications with the Arduino over a serial connection. The second task communicates with the PixHawk through serial communication. By sending commands directly to the Nuttx shell running on the PixHawk, this task can initiate the calibration process on the drone.The python script is also able to read the feedback from the Nuttx shell to determine if the system is ready for the next step in the calibration process. By doing this, we are using MAVLink and not MAVSDK. The third task, Master Mind is the most complex and we will be discussing it in depth.

Figure 20. GUI on Linux Computer

Overall, the Master Mind task is organized in 13 states, which can also be seen in Appendix J. Each sensor in the flight controller has two corresponding states, an orient state and a hold state; each orient state has substates corresponding to the different orientations the system must reach before completing calibration. The messages displayed in the GUI that correspond to the calibration are set in the initialization substate. Master Mind then transitions to the hold state, where it sends the appropriate message to start the sensor calibration on the PixHawk. Once the calibration has been started, Master Mind continues to read the messages from the PixHawk until it receives a confirmation that a sensor is done calibrating, or a side is done. It will then transition back to the orient state from there either increment substates or move on to the next sensor. This repeats for each of the four sensors. The states and substates are largely controlled by the spin and readPX4 function. The entire python code can be found in Appendix I.

There are several functions created for Master Mind. The spin function sends a command to the Arduino to initiate a movement on the calibration machine. This spin function has some additional functionality that affects the transitioning between states. If the spin function is called while the Arduino is not in state 1, it will set a global flag to true, indicating that the Arduino has started a movement. If the movement flag is true and the Arduino is back in state 1, this indicates that the machine has just finished a movement. In this case, if the spin function is called it increments the state of the Master Mind, increments the substate of the system, and resets the relevant flags to prepare for the system to send a new movement to the Arduino. This function takes in the current substate and the desired Arduino movement as inputs.

The readPX4 function interprets the global variable PXmsg. It takes no arguments but simply changes the states of the system based on the messages received from the drone. If the message from the PixHawk begins with a warning or error, it will display the appropriate message and reset the program. Otherwise, if the message is relevant to the progress of the calibration, it will change the Master Mind state and set the appropriate flags. In these cases, there are several options for what the message could be. The string “progress <100>” indicates that the current calibration has been completed. In this case the done flag will be set to true and the state will be decremented by one if Master Mind is in the “hold” stage. The “side done” message acts largely the same as the “progress <100>” message but sets the sideDone flag rather than the done flag. This message corresponds to the magnetometer and accelerometer calibrations, as they must be calibrated on multiple sides before being considered fully calibrated. If the message contains the string “7 s”, the PixHawk is prompting the user to spin the drone for seven seconds. This message only appears when calibrating the magnetometer. The flag spinReady is set, indicating that the primary axis is actively spinning, and Master Mind waits in the spin state before moving back to the orient state to reorient. More details and troubleshooting tips are included in the Operator’s Manual in Appendix K.

5.4 Preliminary and Design Tool Calculations

To ensure that our motors were properly sized and that our structure would not bend excessively or break, we completed torque and bending calculations. We started these on paper so we could have sketches, free body diagrams and the equations written out, then we programmed a design tool in Excel to run these calculations so that we could change numbers easily to test and modify different theoretical possibilities. The hand calculations and design tool screen shots can be found in Appendix E.

For torque calculations and motor sizing, the design tool was very helpful because it allowed us to manually iterate through different combinations of weights and sizes of the innermost motors and see how that would affect the sizing of the corresponding outer motors. With this we were able to properly size our motors due to their weight and torque outputs for the desired RPM (revolutions per minute) of each axis.

We had structural concerns as well that we wanted to check on paper before buying all of materials for a structural prototype. We needed to make sure that the hex shafts on the secondary axis (connecting the motors to the bracket on the ring) weren't going to bend and that the 8020 that makes the base of the fork wouldn't flex either. These were done with two similar calculations which used the respective material properties and inertia calculations to find max bending for both cases. We used a design factor of two in order to ensure our design will operate safely. The maximum deflection that will be experienced by the hex shaft is 0.0042mm and the maximum deflection of the bottom of the fork is 0.39mm. We were comfortable with both of these numbers, so we decided to go ahead with our design.

5.5 Safety and Maintenance

In order to ensure our device is safe and usable we completed a failure mode and effects analysis (FMEA). From the FMEA we determined the most likely and catastrophic failures and developed plans that would ensure the likelihood and impact of these failures would be minimized.

Based on the likelihood of failure and the severity of the failure, our highest priority was ensuring that vibrations do not affect the calibration of the drone. The potential causes of failure are a weak connection between the drone and the mounting plate, attachments that are too flexible, or an incorrectly loaded drone. The action that we will take to prevent these issues is prototyping and testing the drone mount and analyzing the data collected from the calibration logs. In the event that a component fails, we expect to replace it, as most parts are off-the-shelf. If the 8020 beams flex too much under the required loads, we will add reinforcement to the structure, using the T-slot connections. Other safety considerations and preventative measures can be found in the FMEA attached as Appendix L.

Components were chosen to be low maintenance to limit the amount of work required to keep the device functional. Further information about using and maintaining the drone calibrator are included in the Operator's Manual in Appendix K. The only expected maintenance is checking the structural joint fasteners to verify they have not come loose over time.

5.6 Cost Summary

We have two documents detailing cost summary. The first is the Indented Bill of Materials (iBOM), which organizes the parts by the various subsystems in which they are used. The price of certain parts are included, but the overall sum does not completely reflect the total cost as tax, shipping costs, and prices of small parts such as fasteners are not included. The second budget associated with this structural prototype is called the Project Budget and is essentially a shopping list that we used when we ordered all of our parts for manufacturing the structural prototype. This budget sheet contains all the purchases made to construct the final design, and is the most accurate representation of the total cost of the project. This budget shows the final cost is $3,290.44 which is below the $4,000 budget given at the beginning of this project. The current iBOM can be found in Appendix M and the project budget in Appendix N.

6.0 Manufacturing

This section describes our plan for manufacturing and assembling the design specified in Section 5. It contains details on the procurement and assembly of purchased components, as well as the machining of custom manufactured parts.

6.1 Procurement

To mitigate risks in manufacturing, we opted to make the verification prototype design out of as many off-the-shelf parts as possible. By using an 8020 T-slotted rail system, the design is very adjustable and modular. The goal was that if something did not work properly, we could swap it out for another modular part or assembly.

The companies we sourced parts from for the structural prototype were StepperOnline, McMaster-Carr, WCP, 8020.net, and Amazon. Appendix N is a prototype budget showing every part, vendor, and vendor item number needed to order the parts and materials necessary to build the prototype. Included in Appendix N is also a table of links to where each component was sourced.

6.2 Manufacturing Instructions

We tried to keep the components that needed to be physically manufactured relatively small because custom machining and manufacturing is a large investment of time and can also be expensive. All major manufacturing of parts was done in the Cal Poly ME Machine Shops and at Inspired Flight, using equipment such as the waterjet, mills, lathes, and 3D printers. As mentioned above, the modularity of the prototype uses mostly off-the-shelf parts. Besides the cutting of the frame components, the only custom parts in the design were the connections between the frame and various components.

Appendix O is a drawing package that includes all drawings needed for both manufacturing and assembly. Within the drawing package are part drawings for 15 custom machined parts. These parts are as follows:

Table 9. Custom Manufactured Parts

Part Number	Description
207	Slip Ring ½” Hex Shaft
901	Standard ½” Hex Shaft
902-1	Nema 17 Motor Mounting Plate
902-2	Bearing Mounting Plate
902-3	Slip Ring Bearing Support Plate
902-4	Inner Ring Shaft Mounting Plate
902-5	Turn Table Mounting Plate Base
902-6	Turn Table Mounting Plate Fork
902-7	Flanged Hub Adapter
403	Plate Mounting Bracket
914-1	Base Proximity Sensor Mount V1
914-2	Fork Proximity Sensor Mount V1
914-3	Ring Proximity Sensor Mount V1
914-4	Driver Mounting Plate
914-5	Primary Motor Mounting Plate
106	Primary Shaft
402	Drone Mounting Plate
108	Adapter Bushing 12-14mm

With part drawings in hand, follow these instructions to manufacture each part.

<u>Slip Ring ½” Hex Shaft: 207</u>
a) Start with 18” length of ½” hex shaft (ordered from WCP)
b) Cut length of shaft to 4.5” on a chop saw or vertical saw
c) Turn outside of shaft on lathe according to dimensions on drawing
d) On the lathe, drill hole into end of the same side of shaft according to dimensions on drawing

<u>Standard ½” Hex Shaft: 901</u>
a) Start with 18” length of ½” hex shaft (ordered from WCP)
a) Cut length of shaft to 4.5” on a chop saw or vertical saw
b) On the lathe, drill hole into end of the same side of shaft according to dimensions on drawing

<u>Nema 17 Motor Mounting Plate: 902 - 1</u> *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12”x12” sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (engineering drawing dimensions in inches)
c) On a drill press, re-drill the four outer holes to the dimensions shown on drawing
d) On a drill press, re-drill the inner four-hole pattern to the dimensions shown on drawing
e) On a drill press, countersink the inner four-hole pattern to the dimensions shown on drawing

<u>Bearing Mounting Plate: 902 - 2</u> *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12”x12” sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (engineering drawing dimensions in inches)

c) On a drill press, re-drill the four outer holes to the dimensions shown on drawing
d) On a mill, cut down the material directly around the inner hole by the depth listed on the drawing. Note: if done manually, do not worry about overshooting the specified milled profile, as it merely provides clearance for the bearing's flange

Slip Ring Bearing Support Plate: 902 - 3 (*Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12"x12" sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (engineering drawing dimensions in inches)
c) On a drill press, re-drill the four outer holes to the dimensions shown on drawing
d) On a mill, cut down the material directly around the inner hole by the depth listed on the drawing. Note: if done manually, do not worry about overshooting the specified milled profile, as it merely provides clearance for the bearing's flange

Inner Ring Shaft Mounting Plate: 902 - 4 *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12"x12" sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (engineering drawing dimensions in inches)
c) On a drill press, re-drill the four outer holes to the dimensions shown on drawing
d) On a drill press, re-drill the inner four-hole pattern to the dimensions shown on drawing
e) On a drill press, countersink the inner four-hole pattern to the dimensions shown on drawing

Turn Table Mounting Plate Base: 902 - 5 *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12"x12" sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (dimensions in inches)
c) On a drill press, re-drill the six outer holes to the dimensions shown on drawing
d) On a drill press, re-drill the six-hole ring pattern to the dimensions shown on drawing
e) On a mill or by hand, ream the large inner hole to the dimensions shown on drawing

Turn Table Mounting Plate Fork: 902 - 6 *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12"x12" sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (dimensions in inches) using a .dxf file
c) On a drill press, re-drill the four outer holes to the dimensions shown on drawing
d) On a drill press, re-drill the small four-hole pattern to the dimensions shown on drawing
e) On a mill, ream the inner-center hole to the dimensions shown on drawing

Flanged Hub Adapter: 902 - 7 (*Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 12"x12" sheet of ¼" thick aluminum (ordered from McMaster)
b) Cut plate on waterjet cutter (dimensions in inches) using a .dxf file
c) On a drill press, re-drill the two outer holes to the dimensions shown on drawing

Plate Mounting Bracket: 403 *(Note: Cut out multiple plates at once to maximize use of material)*
a) Start with 3"x1"x1.7" block of aluminum
b) On a mill, center drill both four-hole patterns to the dimensions shown on drawing.
c) On a mill, drill the G-sized hole pattern to the dimensions shown on drawing
d) On a mill, drill the #28-sized hole pattern to the dimensions shown on drawing

e) On a mill, chamfer the top and side edges of the part

Base Proximity Sensor Mount V1: 914 - 1

a) Start with a 3D printer and polycarbonate filament (ordered from Amazon)
b) 3D print part using .stl file
c) Ream all 8020 .275" holes with a .275" drill bit
d) Ream all 1/8" holes with a 1/8" drill bit
e) Using a larger drill bit, lightly counter-sink the mounting side of the 1/8" holes for easier assembly

Fork Proximity Sensor Mount V1: 914 - 2

a) Start with a 3D printer and polycarbonate filament (ordered from Amazon)
b) 3D print part using .stl file
c) Ream all 8020 .275" holes with a .275" drill bit
d) Ream all 1/8" holes with a 1/8" drill bit
e) Using a larger drill bit, lightly counter-sink the mounting side of the 1/8" holes for easier assembly

Ring Proximity Sensor Mount V1: 914 - 3

a) Start with a 3D printer and polycarbonate filament (ordered from Amazon)
b) 3D print part using .stl file
c) Ream all 8020 .275" holes with a .275" drill bit
d) Ream all 1/8" holes with a 1/8" drill bit
e) Using a larger drill bit, lightly counter-sink the mounting side of the 1/8" holes for easier assembly

Driver Mounting Plate V1: 914 - 4

a) Start with a 3D printer and polycarbonate filament (ordered from Amazon)
b) 3D print part using .stl file
c) Ream all 8020 .275" holes with a .275" drill bit
d) Ream all 1/8" holes with a 1/8" drill bit
e) Using a larger drill bit, lightly counter-sink the mounting side of the 1/8" holes for easier assembly

Primary Motor Mounting Plate: 914 - 5

a) Start with a 3D printer and polycarbonate filament (ordered from Amazon)
b) 3D print part using .stl file. Engineering drawing also has dimensions
c) Clean up by hand with a de-burring tool and sandpaper

Primary Shaft: 106

a) Start with 300mm length of 15mm diameter steel keyed shaft (ordered from McMaster)
b) Cut length of shaft to 114.30mm (or 4.5") on a chop saw or vertical saw
c) On a lathe, turn down an 89.3mm length of the shaft to 12.70mm
d) On a lathe, turn down the chamfer on the large diameter side of the shaft *(Note: we used the preexisting chamfer on the shaft that came from the manufacturer)*
e) Fit custom sleeve over the 12.70mm end
f) Insert key into keyway and custom adaptor bushing (108)

Drone Mounting Plate: 402

a) Start with provided 24" x 48" x .5" sheet of polycarbonate
b) Use provided .dxf file to waterjet the plate cutouts. (*Note: If aircraft has changed shape, discuss making a new cutout/.dxf with IF engineers.)*

c) Ream all .275” holes with a .275” drill bit

Adapter Bushing 12-14mm: 108

a) Start with a 12-14mm adapter bushing (ordered from McMaster)
b) Drill out the inside diameter to 12.7mm using a mill
c) To cut out the key slot, clamp the bushing on the flat ends between parallels in a mill and use a 5mm end mill to cut a thru slot all the way down the length of the bushing

The drawing package in Appendix O also contains a drawing that lists the dimensions to cut each length of 8020. Table 10 is a table of each part and length required for the prototype:

Table 10. 8020 Cut Lengths for Verification Prototype

Part Number	Description	Length	Quantity
8020-1	Turntable Horizontal Supports	24 in	6
8020-2	Turntable Horizontals long	22 in	4
8020-3	Turntable Horizontals short	20 in	4
8020-4	Turntable Verticals Supports	8 in	4
8020-5	Corner Verticals Supports	9 in	4
8020-6	Fork Arms	33.5 in	4
8020-7	Fork Base	50.5 in	2
8020-8	Ring Motor Supports	3.0 in	4
8020-9	Ring Arms	42.5 in	4
8020-10	Fork Corner Supports	12 in	4

6.3 Assembly

The drawing package (Appendix O) also contains the exploded drawings needed for assembly. With these drawings in hand, and optionally the SolidWorks model open, follow the procedure for each subsystem to assemble the prototype:

Assemble the Fork

Parts required: 505, 904, 901, 903, 902-1, 902-3, 207, 902-4, 905, 906, 914-2, 8020-6, 8020-7, 8020-10, 801, 802, 805, 806, 804

a) Acquire the two bottom rungs of the fork (8020-7)
b) Attach them together with a 3x1” gusset bracket (804) and T-nuts and bolts on the top edge at both ends of 8020 lengths
c) Acquire the four arm sections of 8020 (8020-6)
d) Attach one to each end of the two bottom rings at a right angle. Use the T-nuts and bolts to secure them to the 3x1” gusset brackets already in place. Also add four frame corner fasteners (801), one to the exterior side of each segment joint. Note that the T-nuts may have to be slid in to place before all bolts have been completely tightened (especially for step g).
e) Add a custom Nema 17 motor mounting plate (902-1) at the top edge of the exterior face of each side of the fork (2 plates total) with T-nuts and bolts. Ensure the counter sinks are facing toward the frame structure.

f) Add a custom slip ring bearing support plate (902-3) at the top edge of the interior face of each of side of the fork (2 plates total) with T-nuts and bolts. Ensure the counter sinks are facing toward the frame structure and the cable routing holes are oriented as shown in the exploded view.
g) Add the Frame Corner Supports (8020-10) and Diagonal Brace Ends (807), attached with T-nuts and bolts, to inside the bottom corners of the fork's frame structure as 45° diagonal supports.
h) Add fork proximity sensor mount V1 (914-2) with T-nuts and bolts to the side of the structure.

Assemble the Ring

Parts Required: 505, 904, 901, 903, 902-1, 902-2, 403, 914-3, 8020-9, 8020-8, 803, 801, 905, 906

d) Acquire the four side lengths of the ring (8020-9).
e) Attach together in a square using the 2" gusset brackets (802), T-nuts, and bolts. Ensure the corners are staggered correctly to create a true square and not a rectangle (A single length of 8020 should on have one other piece butted up to its cut end, the other end should be open for T-nut insertion.
f) Add four frame corner fasteners (801), one to each corner with T-nuts and bolts. Ensure that the appropriate number of T-nuts have been put in the slot before tightening as two of the T-nuts for each corner must be slid in from the opposite end of the 8020 length.
   g) Flip the ring over and add four more frame corner fasteners (801) in the same manner.
h) Attach two inner ring shaft mounting plates (902-4) to non-adjacent sides of the ring and secure the hex shafts from the fork to the ring using the hex clamping hubs (903).
i) Drill a 9/16" hole in two of the 8020-9 parts at the midpoint of the ring edge (not the center of the 8020 length). These will be adjacent to the lengths mentioned in part e). This will be for the hex shafts to pass through so they can attach to the bearing mounting plates (902-2) on either side.
j) Tap each end of the all four ring motor supports (8020-8) with an ¼" - 20 tap. Then use frame corner fasteners (801) to attach two ring motor supports (8020-8) above and below the midpoint holes on the side lengths (8020-9). Do this on both sides of the ring.
   k) Secure the bearing mounting plates (902-2) on each side of the ring motor supports (8020-8).
l) One side of the ring will have a Nema 17 motor mounting plate (902-1) instead of the outermost bearing mounting plate (902-2) to allow the Nema 17 motor (505) to be mounted.
m) Attach the two hex shafts (901): one hex shaft to the bearings in the bearing mounting plates (902-2) on one side of ring and the other hex shaft to the opposite side of the ring to the Nema 17 motor (505)
n) Add ring proximity sensor mount V1 (914-3) with T-nuts and bolts to the side of the structure

Assemble the Base

Parts Required: 506, 102, 103, 104, 105, 106, 107, 108, 109, 902-5, 902-6, 902-7, 909, 910, 911, 914-1, 914-4, 914-5, 8020-1, 8020-2, 8020-3, 8020-4, 8020-5, 802, 110

a) Acquire the 8020 lengths required (8020-1, 8020-2, 8020-3, 8020-4, 8020-5) and piece together the base structure attaching with 16 frame corner fasteners (801), 24 gusset brackets (802), and needed T-nuts and bolts. Ensure that the appropriate number of T-nuts have been put in the slot before tightening as some of the T-nuts for each corner must be slid in from the opposite end of the 8020 length
b) See CAD and assembly drawings in appendix E for how 8020 components fit together. Make sure to slide in extra T-nuts for the base proximity sensor mount V1 (914-1) and the driver mounting plate (914-4)

c) Tap the bottom side of the corner support verticals (8020-5) and attach the leveling feet (110) to them.
d) Assemble the primary drive shaft and turntable section (Reference *Figure 15. Primary Motor Torque Conversion*)
   i. Acquire turned down 15mm keyed primary shaft (106) and attach one side into the 15mm x 15mm coupling (109) and then thread it through the slip ring.
   ii. Attach the turntable (103) to the turntable mounting plates (902-5, 902-6) with T-nuts and bolts.
   iii. Pass the shaft through the turntable (103) and turntable mounting plates (902-5, 902-6) and attach it to the adaptor bushing (108) and the keyed flanged hub (105) and key (107).
e) Attach this shaft/turntable subassembly to the long turntable horizontals (8020-2) with the turn table mounting plate base (902-5) and T-nuts and bolts.
f) Acquire the primary motor mounting plate (914-5) and attach the Nema 34 motor (506) to it with M5 screws (911).
g) Attach this to the lower set of long turntable horizontals (8020-2) and insert the shaft into the other side of the 15mm x 15mm coupling (109)
h) Attach the flanged hub adaptors (902-7) to the top of the turn table mounting plate fork (902-6)
i) Add base proximity sensor mount V1 (914-1) with T-nuts and bolts to the side of the structure
j) Add driver mounting plate (914-4) with T-nuts and bolts to the side of the structure

Assemble the Drone Mounting Assembly

Parts Required: 401, 901, 402, 403, 404, 912, 913, 805, 903, 905

a) Acquire the custom drone mounting plate (402) and attach the straps (401) and bubble level (404) to it with 1/2" wood screws (919).
b) Attach both plate mounting brackets (403) to the pre-drilled mounting holes in the custom plate (402) with ¼" – 20 bolts (805) and ¼" locknuts (913).
c) Attach hex clamping hub (903) to the plate mounting brackets (403) with the 6/32 2" screws (905).

Final Assembly

Parts Required: 100, 200, 300, 400

a) Attach the fork assembly (200) to the base assembly (100) using T-nuts and bolts, from the fork base (8020-7) to the turn table mounting plate fork (902-6).
b) Attach the ring assembly (300) to the fork assembly (200) by coupling the hex shafts (901) on the fork (200) and the hex clamping hub (903) on the ring (300).
c) Attach the drone mount (400) to the ring assembly (300) by coupling the hex shafts (901) on the ring (300) and the hex clamping hub (903) on the drone mount (400).

6.4 Discussion of Lessons Learned and Recommendations

Throughout the process of manufacturing, we encountered challenges and learned a lot about how to best manufacture a product with many off-the-shelf and custom-made components. Because we tried to make everything out of 8020 T-slotted aluminum, our design was nicely modular and easy to assemble. Working with 8020 was forgiving and simple. It was occasionally time consuming to screw all the fasteners in when assembling or modifying the structure, but the strength, modularity, and flexibility in design made it worth it.

While we wanted to use as many off-the-shelf components as possible, we still had to manufacture some custom parts ourselves. As seen in Table 9, we had about fifteen custom manufactured components. Most of these were mounting plates that were cut out of ¼" 6061 aluminum using the waterjet in Mustang 60. We discovered near the end of manufacturing, however, that we were able to 3D print these plates using a polycarbonate blend of filament, which dramatically decreased the amount of manual machining while achieving similar strength properties. Using a waterjet to cut aluminum plate was very simple and effective, mostly because it was freely available for students. However, it did require additional machining on a mill to make sure the holes were sized and chamfered properly. After seeing the strength of the 3D printed polycarbonate parts, we think it would have been more efficient for all plates on this prototype to have been 3D printed. Further lessons learned and suggestions for the future are included in Section 9.0 Conclusions and Recommendations.

7.0 Design Verification

This section details the process we intended to follow to test the verification prototype with the engineering specifications stated in Section 3.3 of this report, as well as the specifications we were able to determine by inspection. A summary of the tests will be described below and the detailed plan is shown in the Design Verification Plan (DVP) spreadsheet as Appendix P.

7.1 Engineering Specifications

To reference the engineering specifications, see Table 4 in Section 3.3. Most of the specifications can be verified by simple inspection, for example, quickly measuring the dimensions of the base or determining the number of steps required to set up and run the device.

There are only two specifications which require planned testing procedures. The full procedures can be found in Appendix Q. These are the positional accuracy of the device and the reliability. To test the positional accuracy, we plan to attach a pre-calibrated IMU to the drone plate and run a series of step responses controlling the device to reach one of the six orientations. We will determine the accuracy by finding the error between the desired value that we input, and the actual position as measured from the IMU. We plan to run multiple trials for each orientation, as well as test all six orientations. We will determine the reliability by running a drone through a complete calibration multiple times. We will record the number of calibrations that are successful, and the number that result in errors. The percentage of successfully calibrated drones will be considered the reliability of the device. To see basic descriptions of the testing procedure for the rest of the specifications, refer to Section 3.3.

7.2 Verifying Specifications

Due to project delays with completing the final functional prototype, and debugging the software controlling the device, we were unable to complete as much testing as we had hope to verify that the design met each of the specifications determined at the start of the project. However, a large number of our specifications are met by simple inspection testing and did not require an in-depth testing procedure to determine if the design met the criteria. A modified specifications table is shown in Table 11 that states whether the specifications were met, not met, or if we were unable to determine the result. These are shown in the table with the symbols P, for pass, F, for fail, and N/A, for unable to determine in the time remaining. A full assessment of the risks, including the risks of not satisfying these requirements, can be found in Appendix R.

Table 11. Modified Specification Table with Results

Spec. #	Specification Description	Requirement or Target (units)	Tolerance	Risk	Compliance	Result
1	Accurate drone calibration	±1 degree	Max	H	T	N/A
2	Number of set up steps	5	Max	M	I	P
3	Time to run	10 min	Max	M	I	P
4	Compatible drones	IF current and future drones	Y/N	M	I	P
5	Budget	$4,000	Max.	M	I	P
6	Weatherproof	Can be used outside	Y/N	M	I	F
7	Reliability	99%	Min	M	T	N/A
8	Size	4' x 4' base	Max	L	I	P
9	Lifetime	1,000 uses	Min	L	A	N/A
10	Weight	50 lbs.	Max	L	I	N/A
11	Fits through door	32" door frame	Y/N	L	I	P
12	Standard parts	When possible	Y/N	L	I	P
13	Uses wall power	120 V	+/- 5V	L	I	P

The specifications that have a N/A result are mainly because we did not have access to the tools required to perform the test or inspection, or they are a specification related to the performance of the project which as stated before, was not fully functional in time to allow for testing. The specification that failed is the device being Weatherproof since the electrical hardware including the Arduino microcontroller, and motor drivers are uncovered. Any sort of precipitation could damage these components so the device should not be used in inclement weather. We were aware of this in the design, but once we determined that the calibration did not need to be completed outside, it became a less important.

The specifications that are marked as passing were determined by simple acts or inspections by team members. Examples of this include us carrying the device through a doorway with two people with relative ease, determining the number of steps in the set up procedure from the user manual, and knowing that the device is made of mainly standard parts and is powered using a standard wall outlet. Some of the inspections were more qualitative such as allowing compatibility for future drones. We determined this specification to be met because the only part that must be changed for future drones is the attachment plate, and a new cut out could be made to fit the any drone's geometry. This plate can be removed, and a new one screwed on in its place without making any other changes to the device. This requires minimal work and new parts, therefore the design meets that specification.

Since we were unable to complete all of testing we had deemed necessary to verify the design met the specifications, we recommend this be one of the next steps carried out by Inspired Flight in order to improve upon this project. This is one of the major tasks included in the Section 9.3 .

8.0 Project Management

The Project Management section outlines the steps that our team plans to take in order to meet the goals set for this project. It explains the process we will take from initial research and conceptual design, all the way to prototyping the final design and testing the effectiveness. A Gantt chart is included as an

Appendix S to show the timeline and schedule for different sections of this project. Finally, there is a plan for testing that will be completed to determine if the specifications were met.

8.1 Overall Design Process

- Defining
  We will first need to define the problem. Refining the scope of the project is a byproduct of defining the problem, as design considerations will refer directly to the problem.
- Brainstorming
  We will then brainstorm hundreds of designs and sub-designs as a group. Categorizing these ideas by importance using a Pugh Matrix or other similar method will help narrow down the perfect design.
- Managing
  Before moving forward, we will need to make projections of what we expect our design choices will entail. This will include detailed planning, tracking, and controlling of team activities, expenses, workload, and risks involved in bringing the project to fruition. This step will continue throughout the rest of the project.
- Prototyping
  Once the design is on paper, we will move forward strategically. We will first create a prototype that fulfills as many requirements as possible. Further iterations will be implemented as needed. This should be a happy-medium between cost and functionality, usually on the cheaper side.
- Testing
  Analyzing the prototype as extensively as possible will be important, as any design flaws will be much cheaper to fix if addressed as early as possible. If needed, we will then revert to a previous step (e.g. brainstorming) and repeat the rest of the process as necessary for each iteration.
- Reviewing
  The design process includes three major mile markers in the form of design reviews. The first is the Preliminary Design Review (PDR), then the Critical Design Review (CDR), and lastly the Final Design Review (FDR). These designate times to update Inspired Flight with progress and help the team collect the work already done and refocus before moving forward.
- Delivering
  After all the necessary iterations have been made, the final product will be produced. This product should satisfy all the requirements listed in the problem and should balance the needs and wants in a way that best satisfies the project management.
- Benchmarking
  The final step is to confirm the performance of the product really does satisfy the problem needs. This could be in the form of cycle-testing, surveying users, etc.

Table 12. Key Project Deliverables/Dates

Deliverable	Date	Quarter
Scope of Work	10/18/19	Fall
PDR Presentation	11/12/19	Fall
Submit PDR	11/15/19	Fall
Interim Design Review	1/16/20	Winter
CDR Presentation	2/4/20	Winter
Submit CDR	2/7/20	Winter
Manufacturing & Test Review	3/12/20	Winter
Submit FDR	5/28/19	Spring
Project Expo	5/29/20	Spring

8.2 Unique Prototyping/Testing Techniques for Project

Analyzing PixHawk Sensor Outputs:
The onboard flight controller has the capability of recording vast amounts of data (direct and interpolated) from various sensors within the flight controller. Each power cycle of the drone records a flight log with all said information we will utilize for analysis.

8.3 Next Steps

As seen in the Gantt Chart in Appendix S, the Final Design Review is last deliverable of this project, so our group will be delivering the project and all of our documented work to Inspired Flight, but we will not be continuing to work on this project ourselves. We have provided all of our work should Inspired Flight continue to develop this project internally or have another senior project group design a solution to this problem. Certain tasks which we were unable to complete are described in section 9, and it is our recommendation for these next steps to be taken to continue testing and improving the device.

8.4 Project Management Reflections

Overall, we were very happy with how the project was managed. Jackie did a great job in the role of project manager and kept the Gantt chart very up-to-date and kindly reminded people of tasks and due dates. She also did a great job of making sure the rest of the team was up to date on the general progress of the project. The whole team was responsible with their own tasks and, although the project timeline was adjusted a few times due to lead times and design changes, overall, the project stayed on track. There were minimal team disagreements and ones that did arise were handled maturely and effectively.

We conducted weekly meetings that were outside of class time and we found these to be very helpful. During fall quarter we met weekly as an entire team, allowing us to bond as a team and stay accountable with deadlines that we set. In winter and spring quarters, we split the meetings into mechatronics and mechanics groups. Jackie, Ryan, and Zach met weekly to work on software and electronics. Matthew and Tyler met in the shops/labs to do manufacturing and design.

As mentioned above the project tracked with the timeline pretty well. Some of the contributing reasons that made this possible were the weekly check-ins with our advisor, separate weekly team meetings, and having an up-to-date Gantt chart.

In terms of improvements to the project management, we would have increased communication about specific tasks and delegated the larger tasks more evenly. While it was helpful to separate into two teams to focus on the different aspects of the project, spreading the mechatronics and mechanical work more evenly would have helped team members stay up to date on the various states of the project. Synchronizing schedules and meeting up to work on tasks was helpful to keep everyone involved and we would recommend, especially with a larger team of five, to do more tasks with other team members.

On the manufacturing side, it would have been helpful to have completed the detailed drawings for the custom-made parts sooner. This would have allowed more efficient manufacturing and better design communication amongst team members.

9.0 Conclusions and Recommendations

This senior project, sponsored by Inspired Flight, aims to improve Inspired Flight's drone calibration process on accuracy and required man hours. Ultimately, there is one main goal: to create an improved way to calibrate Inspired Flight drones. The secondary, more lofty goal, is to simulate a test flight in the effort to confirm proper operation of the drone.

Through working closely with Inspired Flight, we determined the engineering specifications found in the Objectives section. These constrain the design and ensure that all needs are met. Additionally, we have developed concept designs and completed analysis to further refine and improve the design. We have completed as much of the manufacturing and final testing as possible under the current circumstances of the COVID-19 pandemic.

In order to evaluate the success of the project, the engineering specifications will be measured in accordance with the compliance method shown in Table 4. The overarching success will be seen by using the design to calibrate a drone, hopefully conduct a simulated flight, and then fly the drone successfully (with no calibration errors).

The purpose of this document was to state the goal of improving calibration of Inspired Flight drones, describe the final design and the completed prototype, and give as much information as possible to help with the next steps of this project, should Inspired Flight want to continue to improve this design. This Final Design Report is intended to inform Inspired Flight of all of the work we completed, and we hope they are satisfied with all that we were able to accomplish.

9.1 Reflection

Looking back on the last year, the calibration system project has been a great learning experience for all of us. We utilized our engineering knowledge to design the project to the best of our abilities, then were able to verify our own design, giving us direct feedback toward the flaws of our design. Having a hand in every step of the process, from ideation to manufacturing, taught us a lot about ourselves. Each team member had strengths and weaknesses, and learning to navigate this dynamic was another valuable takeaway. As mentioned above, the Covid-19 pandemic posed many challenges during the last third of the project. Manufacturing using power tools or the shops at Cal Poly was no longer an option, resources that we thought were essential in finishing the project. Fortunately, we were able to grab the unfinished project out of the Cal Poly labs during spring break and finish the final manufacturing at Inspired Flight. Because the calibration system required heavy programming, most of the physical design was finished before the start of the pandemic. This helped us to see the light at the end of the tunnel, since coding the system was much easier to do remotely.

What we accomplished was very close to our goal, and we consider the project a success. As stated, the goal was to have a device that one of Inspired Flight's drones could mount into, and with the click of a button could calibrate all sensors on the aircraft. Apart from a slight hiccup in the stability of the secondary motor, the system achieves this goal at a basic level. This and similar issues are discussed below.

Motor aside, the calibration system is still very much a prototype. Given the very limited amount of time and resources we had to complete the project, we recommend treating the system as a proof of concept. We hope to provide enough information and lessons learned that Inspired Flight can smoothly pick up where we left off.

9.2 Issues and Recommended Fixes

Below is a list of the lessons we learned from designing and building the drone calibrator.

Physical Design:

- The hex c-clamps do not tighten very well, so they tend to allow the hex shafts to slip. Tight tolerancing was needed to allow for tightening. Replace them with a different solution.
- The button heads of the screws holding the flanged hub to the 8020 fork on the primary axis interfered with the primary axis adapter plate. We should have included fasteners in the CAD and that would have made this problem obvious. The best solution to this would be to remove more material in the primary axis adapter plate to allow the button heads clearance. Another fix would be to increase the thickness of the flanged hub adapter (902-7). This is effectively what we did by adding washers in with the shims. This, however, required longer screws than anywhere else in the assembly (1/4"-20 1") and it decreased contact where the fork is held on the primary axis. Note that the current CAD model has a thicker flanged hub adapter (902-7)
- The first attempt at a primary motor mounting plate was impossible to assemble because it required bolts coming in from both sides. When putting in one set of bolts, the motor then blocked the holes for the 8020, and the same issue occurred if assembled in the opposite order. The current plate fixes this design flaw, and we should have modeled the fasteners in CAD and considered the assembly of the structure during the design stage.
- The 12-wire slip ring (102) that routes through the primary axis is being held in place through tension in the wires. This is not a long-term solution, but we did not realize this issue until it was too late for us to design a good solution for it. This should not be too difficult of a design, but we would recommend finding a solution for long-term use. An idea that was discussed was a standoff that connects the slip ring (102) to the primary motor mounting plate (914-5).

Manufacturing and Assembly:

- We water-jetted most of our aluminum plates. Then, due to the COVID 19 pandemic, machining the last few this way was no longer an option. Instead we 3D printed them with polycarbonate filament. If we were to redo this project, we would 3D print all plates. It would save time and money, as the parts would need neither custom cutouts nor extensive manual machining. However, if this were to be mass-manufactured, aluminum plates may be more time and cost effective, but a cost analysis should be conducted to determine the less expensive option.

- For testing, we used set screws to secure the hex shafts on to the motor shafts. When we tried to disassemble, we sheared the set screw. This caused us to waste three hours trying to remove the partial set screw. We would recommend using just JB Weld instead of set screws. This is what is used on the tertiary motor. However, the secondary motor still has the original set screw since the motor itself needs replacing.
- It was nearly impossible to perfectly line up the secondary axis with the center of the ring. When we tried, we realized that this shifts the weight off-center, requiring more torque from the secondary motor to rotate. The best solution we found was to load the drone and then balance the ring by sliding the secondary axis relative to the 8020 ring arms.

Electronics:

- The biggest issue with operating the prototype as delivered is the secondary motor. It appears that it was damaged electrically during testing. We believe that the damage was caused while running a higher current through it than it was rated for. This was due to our original plan to drive two stepper motors using a single, higher-powered motor driver, however controlling both motors using this method did not work well. Once the project's functionality has been deemed satisfactory, we recommend swapping out this Nema 17 with an identical replacement.
- While testing, we found that there is electrical noise in the signal travelling long distances, notably to the Reed switches. To combat this, we made our software more robust to sift through the electrical noise.

Software:

- We spent a long time attempting to use MavSDK to run the calibrations. However, we realized that there was not a function to calibrate the level horizon. This led us to find a new solution using just MAVLink.
- We had trouble getting consistent results running the accelerometer calibration especially. To fix this we updated the MAVLink firmware.
- Due to time constraints, we needed to write much of our code without the whole project assembled to test it on. This made it difficult to anticipate the bugs in our code. To combat this, we should have pushed up the timeline to have more physical, testable hardware while the software is being developed.
- There was a bug where the tertiary motor would begin moving as soon as Master Mind entered the accelerometer calibration stage. This was because the primary motor was still moving when the calibration began, so the accelerometer calibration thought it was further along then it was.

9.3 Looking Forward

This project was difficult to complete due to the COVID-19 pandemic shutting down many things, including the Cal Poly machine shops and in person meetings. Even so, we accomplished almost everything we set out to do. That being said, there are a number of things which we wish we could have done before handing the project over; these are listed below.

- As mentioned in the electronics section in 9.2, we highly recommend replacing the secondary motor. The motor seems to be burnt out, but the specifications are good, so the same model should be fine.

- We originally intended to make the final drone attachment plate out of polycarbonate. Due to COVID-19 closing the machine shops, we had to stick with our wooden test plate. We recommend replacing it with polycarbonate for greater rigidity. Once this plate is manufactured, we highly recommend adding padding for the drone. This raises the center of gravity, optimizing the moment of inertia for the secondary axis. Special focus should be given to making the padding thick enough to allow room for the batteries to connect on the drone without interference. Alternatively, adding cutouts for the battery connectors could be a viable solution.
- We are aware that the micros() function, used for timing interrupts in the Arduino code, overflows after about 70 minutes. Due to lack of time, we were not able to account for this issue in our code nor test what issues it may cause. This is unlikely to cause issues due to the long time period, however, we would recommend adding a catch to the Arduino code to account for the overflow.
- The testing plans that we created still need to be completed. These tests will help determine if the device we designed has the ability to meet the required angular positional accuracy for each axis. We not only ran out of time to test these, but the secondary axis motor not functioning properly would further hinder the test results as it would be unclear if it failed due to overall design or because the motor was faulty.

As this is primarily a proof of concept, we expect that there are more flaws than listed above. With unlimited time and resources to test and iterate upon this design, we are confident there are many improvements possible. However, as it is upon delivery, this drone calibrator successfully fulfills its purpose and meets many of the project requirements.

Works Cited

Burdziakowski, Paweł, and Katarzyna Bobkowska. “Accuracy of a Low-Cost Autonomous Hexacopter PlatformsNavigation Module for a Photogrammetric and EnvironmentalMeasurements.” *Proccedings of 10th International Conference "Environmental Engineering"*, 2017, doi:10.3846/enviro.2017.173.

“Unmanned Aircraft Systems.” *AFD*, California Polytechnic State University, 2019, afd.calpoly.edu/risk-management/drones.

El Shair, Zaid & Rawashdeh, Samir. (2019). Design of an object sorting system using a vision-guided robotic arm.

Gagne, Don. “Overview · QGroundControl User Guide.” *QGroundControl*, GitBook, 25 July 2019, docs.qgroundcontrol.com/en/.

Iqbal, Jamshed & Ul Islam, Muhammad & Khan, Hamza. (2012). Modeling and analysis of a 6 DOF robotic arm manipulator. Canadian Journal on Electrical and Electronics Engineering. 3. 300-306.

Looney, Mark. “A Simple Calibration for MEMS Gyroscopes.” EDN Europe, July 2010.

Mack, G. “A Drone Calibration Turntable.” *DJI Phantom Drone Forum*, 2016, phantompilots.com/threads/a-drone-calibration-turntable.86244/.

Movsesian, Peter. *Mobile Robotic Arm*. 9 May 1995.

Quigley, Morgan, et al. *A Low-Cost Compliant 7-DOF Robotic Manipulator*. Stanford University.

Stollmeyer, Marc & Bialy, Martin. Personal interview. 8 October 2019.

United States, Congress, FAA. “Summary of Small Unmanned Aircraft Rule (Part 107).” *Summary of Small Unmanned Aircraft Rule (Part 107)*, 2016.

Watson, Joshua, et al. *Unmanned Aerial Vehicle Sensor Calibration during Flight*. 24 July 2018.

繆志豪 . “CN107655470A - Yaw Angle Value Calibration Method and System for Unmanned Planes.” *Google Patents*, Google, shorturl.at/dkmW9

智灵飞（北京）科技有限公司 . “CN107894241A - Ellipsoid Fitting-Based Unmanned Aerial Vehicle Magnetic Sensor Calibrating Method and Unmanned Aerial Vehicle.” *Google Patents*, Google, shorturl.at/iqAIO

上海拓攻机器人有限公司 . “CN106153025A - Multi-Rotor Unmanned Aerial Vehicle and Calibrating Method and System for Electronic Compass Thereof.” *Google Patents*, Google, shorturl.at/kuyNR

Appendices

Appendix A: Current Calibration Process and Errors

Step-by-Step of the Current Procedure

1. Open QGroundControl software on computer
2. Power on the aircraft
3. Calibrate compass
    a. Rotate aircraft about its vertical axis in each of its six orientations
4. Calibrate gyroscope
    a. Place aircraft on a non-moving surface
5. Calibrate accelerometer
    a. Hold aircraft still in each of its six orientations
6. Calibrate level horizon
    a. Place aircraft on a level surface

Note: Compass is usually calibrated outdoors due to possible interference encountered indoors

Possible Errors Due to Poor Calibration

- ‘High Accelerometer Bias’ – caused by poor accelerometer calibration
- ‘Compasses Inconsistent’ – caused by improper specifying of compass orientation or poor compass calibration
- Large amounts of drift during manual flight – caused by poor level horizon calibration

Appendix B: Quality Function Deployment (QFD)

Correlations	
Positive	+
Negative	−
No Correlation	

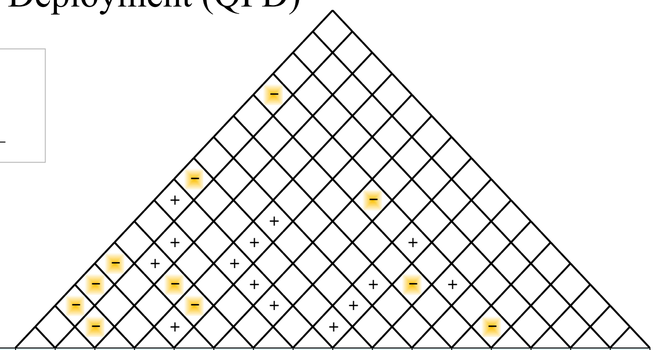
Relationships	
Strong	●
Moderate	○
Weak	▽

Direction of Improvement	
Maximize	▲
Target	◇
Minimize	▼

QFD House of Quality

Project: _Drone Calibration _______

Revision Date: __10/13/19_______________



WHO: Customers | NOW: Curr. Products

Row #	Weight Chart	Relative Weight	Inspired Flight (Company)	Drone Engineer	Maximum Relationship	WHAT: Customer Requirements (Needs/Wants) \ HOW: Engineering Specifications (Tests)	Budget	Size constraints	Weatherproof	Lifetime	Reliability	Weight	Fits through door	Standard parts	Accurate positioning	Number of set up steps	Uses wall power	Time to run	Compatible drones				Our Current Product	By hand	Turntable	Commercial Robotic Arm	Row #
						Column #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16					
						Direction of Improvement	▼	▼	◇	▲	▲	▼	◇	◇	▲	▼	◇	▼	▲								
1		5%	8	2	9	Low cost	●	▽						○	▽									5	4	1	1
2		5%	3	5	9	Compact		●				○	●											5	4	2	2
3		5%	7	3	9	Survive elements			●		○		▽											5	4	1	3
4		9%	8	8	9	Reliable				○	●			○	○									4	5	5	4
5		9%	7	9	9	Transportable		●				●	●											5	4	2	5
6		7%	6	7	9	Robust			●	○	▽													5	3	4	6
7		7%	4	8	9	Easy to maintain			○		▽			●										5	4	2	7
8		8%	9	5	9	Adaptable								○			○		●					5	2	3	8
9		9%	7	9	9	Easy to use						▽				●	○	●						2	3	4	9
10		1%	1	0	9	Power consumption	○		▽								●							5	5	2	10
11		8%	9	5	9	Time savings					▽		▽			●		●						1	3	3	11
12		5%	8	2	9	Accuracy					○				●									1	4	4	12
13		11%	10	10	9	Can calibrate									●									4	3	3	13
14		11%	10	10	3	Can test calibration					○													5	1	1	14
15		0%																									15
16		0%																									16

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
HOW MUCH: Target Values	$4,000	4x4 ft	Weatherproof	1000 Uses	99% Reliable	50lbs or Less	Fits Through 32" Door	Utilizes Standard Parts	Accurate to +/- 0.1 Degrees	5 Setup Steps	Powered off 120v Wall Outlet	10 Minute Run Time	Compatible with Current and Future			
Max Relationship	9	9	9	3	9	9	9	9	9	9	9	9	9			
Technical Importance Rating	49.5	127.6	135.7	48.76	168.1	104	135.4	127.7	180.7	150.1	54.68	150.1	68.86	0	0	0
Relative Weight	3%	9%	9%	3%	11%	7%	9%	9%	12%	10%	4%	10%	5%	0%	0%	0%
Curr. Products: Our Current Product	1	3	4	2												
By hand	2	3	5	1												
Turntable	3	0	4	5												
Commercial Robotic Arm	4	1	5	4												
0	5	5	2	1												
Column #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Appendix C: Stanford Brainstorming Activity

CONSTRAIN MOVEMENT	SUPPORT WEIGHT	MINIMIZE VIBRATIONS	MEASURE ANGLE	APPLY FORCE	MEASURE POSITION	SPIN
goo	table	mass dampers	IMU	motor	hard stop	turtle
hold it	platform	dampers	potentiometer	kick it	encoder	turn table
magnets	strings	damping glue	bubble level	turtle	limit switch	manually
drill a hole	truss structure	goo	visually	gimbal	pixhawk sensors	hamster ball
string	bridge	put on chicken head	plumb line	actuator	visually line up	bicycle
put in cut out (neg of drone)	carbon fiber	gimbal	protractor	pneumatics	computer vision	balanced cage
sit on it	float it	piezo electric	encoder	manually	moved cat bar?	gyroscope-like rig
duct tape	cage	PID	by the moons and stars	use drones own motor power	visually	merry go round
strap down legs	giant balloons	frickin over engineered structure	must balance else falls over	electrical juice	go-nogo gauge	
universal fixture	more drones	elastics	stepper motor steps	worm gear		
kevlar wire	hold it	use px sensors	gear teeth	string pulley		
tie down	trash pile	have drone fight ures?	weighted to be aufo?	hydraulics		
bolt on with drone adapter plate	slack line between two poles			levers		
bolt down	set on ground			with your hand		
zip tie	super glue			linear actuators		
claw	suspend by strings			lead screws		
cage	antigravity			tow it		
hold it and spin with it	balance on a pole			tip it over		
bite it	balance on two poles			shaft		
	balance on a tripod			pinon gear		
	spaghetti					
	carabiner					
	disco ball					

Appendix D: Pugh Matrices

Structure

Criteria	Gyro Gimbal	Camera Gimbal	Turntable Arm	Sliding Half Moon	Hamster Ball on Rollers
Low Cost	0	-1	1	0	0
Compact	0	1	1	0	0
Robust	0	-1	0	-1	1
Manufacturable	0	1	1	1	0
Vibrations	0	-1	-1	-1	1
Total	0	-1	2	-1	2

Attachment

Criteria	Clamp Sandwich	Colt to Adapter Plate	Strap to Plate - Body	Strap to Plate - Legs	Square Key Lock
Low Cost	0	1	1	1	0
Easy to Use	0	-1	0	0	0
Robust	0	0	0	0	-1
Adaptable	0	1	1	1	1
Vibrations	0	1	0	0	0
Manufacturable	0	1	1	1	-1
Total	0	3	3	3	-1

Drive Motion

Criteria	DC + Encoder	Servo	Stepper	Linear Actuator
Low Cost	0	-1	1	-1
Weight	0	0	0	-1
Survive Elements	0	0	0	1
Reliability	0	1	-1	0
Power Consumption	0	1	-1	0
Accuracy	0	0	0	0
Vibrations	0	0	-1	0
Total	0	1	-2	-1

Measurement

Criteria	Encoders	IMU	Potentiometer	Bubble Level	Stepper Motor Steps	Hard Stop
Low Cost	0	0	1	1	1	0
Survive Elements	0	-1	0	1	0	1
Accuracy	0	-1	-1	1	0	1
Reliability	0	-1	-1	1	-1	1
Total	0	-3	-1	4	0	3

Drive Train

Criteria	Direct Drive	Gears	Worm Gear	Belts	Chains
Low Cost	0	-1	-1	-1	-1
Compact	0	-1	-1	-1	-1
Easy to Maintain	0	-1	-1	-1	-1
Robust	0	-1	-1	-1	-1
Efficiency	0	-1	-1	-1	-1
Accuracy	0	-1	-1	-1	-1
Manufacturable	0	-1	-1	-1	-1
Total	0	-7	-7	-7	-7

1/2

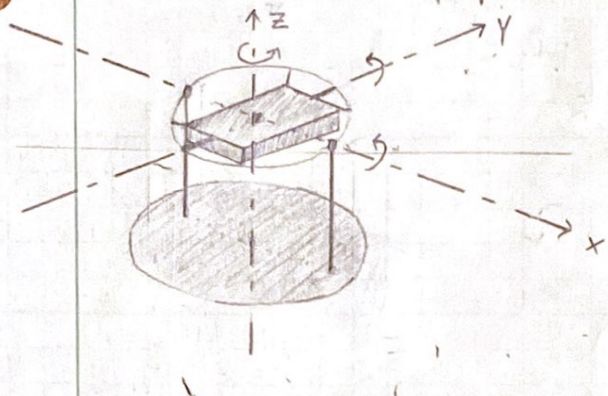
HIGH LEVEL CALCULATIONS

Appendix E: Preliminary Hand Calculations

Max Take-off weight ⇒ $m = 6.2$ Kg

$m = 6.2/2 = 3.1$ Kg

Max Radius of Propellers ⇒ $r = .40$ m



Aprox Drone Dimentions:

0.25m X 0.35 m

Angular Acceleration:

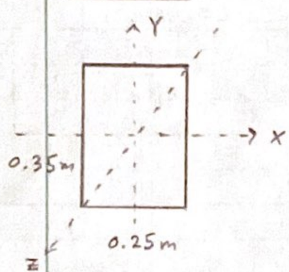
$w = 3 \frac{rad}{s}$

$\alpha = 1.5 \frac{rad}{s^2}$

★ Rather arbitrary

$\tau = I \alpha$

DRONE



$I_{xx} = \frac{1}{12} m \cdot a^2$

$I_{xx} = \frac{1}{12}(3.1kg)(0.35m)^2$

$I_{xx_{Drone}} = 3.16e-2$ [$kg m^2$]

$I_{yy} = \frac{1}{12} m \cdot a^2$

$I_{yy} = \frac{1}{12}(3.1kg)(0.25)^2$

$I_{yy} = 1.61e-2$ [$kg m^2$]

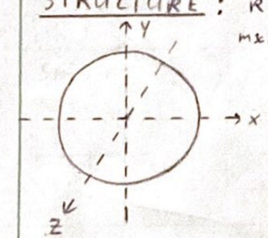
$I_{zz} = \frac{1}{12} m (a^2 + b^2)$

$I_{zz} = \frac{1}{12}(3.1)(0.35^2 + 0.25^2)$

$I_{zz} = 4.78e-2$ [$kg m^2$]

STRUCTURE: Ring (Thin) + Base (Solid)

$m \approx 1kg$, $r = 0.42m$; $M \approx 3kg$, $R = .65m$



$I_{xx} = \frac{1}{2} m r^2$

$I_{xx} = \frac{1}{2}(1kg)(0.4)^2$

$I_{xx_{Ring}} = 0.08$ [$kg m^2$]

$I_{xx_{total}} = I_{xx_{Ring}} + I_{xx_{Drone}}$

$I_{xx_{total}} = (0.08) + (0.0316)$

$I_{xx_{total}} = 0.111$ $kg m^2$

$\tau_{xx\ total} = 0.167$ N·m

About Z-axis

$I_{zz_{Ring}} = m r^2$

$I_{zz_{Base}} = \frac{1}{2} M R^2$

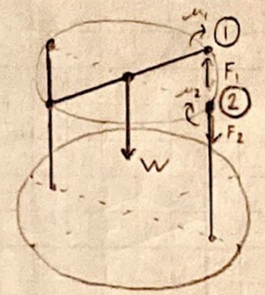
$I_{zz_{Total}} = I_{zz_{Ring}} + I_{zz_{Base}} + I_{zz_{Drone}}$

$I_{zz_{Total}} = (m r^2)_{Ring} + (\frac{1}{2} M R^2)_{Base} + (\frac{1}{12} m (a^2 + b^2))_{drone}$

$I_{zz_{total}} = (1kg)(0.4)^2 + \frac{1}{2}(3kg)(0.65)^2 + (4.78e-2)$

$I_{zz_{total}} = 1.50$ $kg m^2$

$\tau_{zz\ total} = 0.84$ N·m



Aproximate Drone as point mass, W (N)

Assume perfectly ridgid structure, therefore $F_1 = F_2$

$$F_1 = -F_2 = \frac{W}{2} = \frac{(3.1)(9.81)}{2}$$

$$F_1 = 15.2\ N$$

$$M_1 = F_1 \cdot r$$

$$M_1 = (15.2N)(0.4m)$$

$$\boxed{M_1 = 6.08\ N \cdot m}$$

Appendix F: Design Hazard Checklist

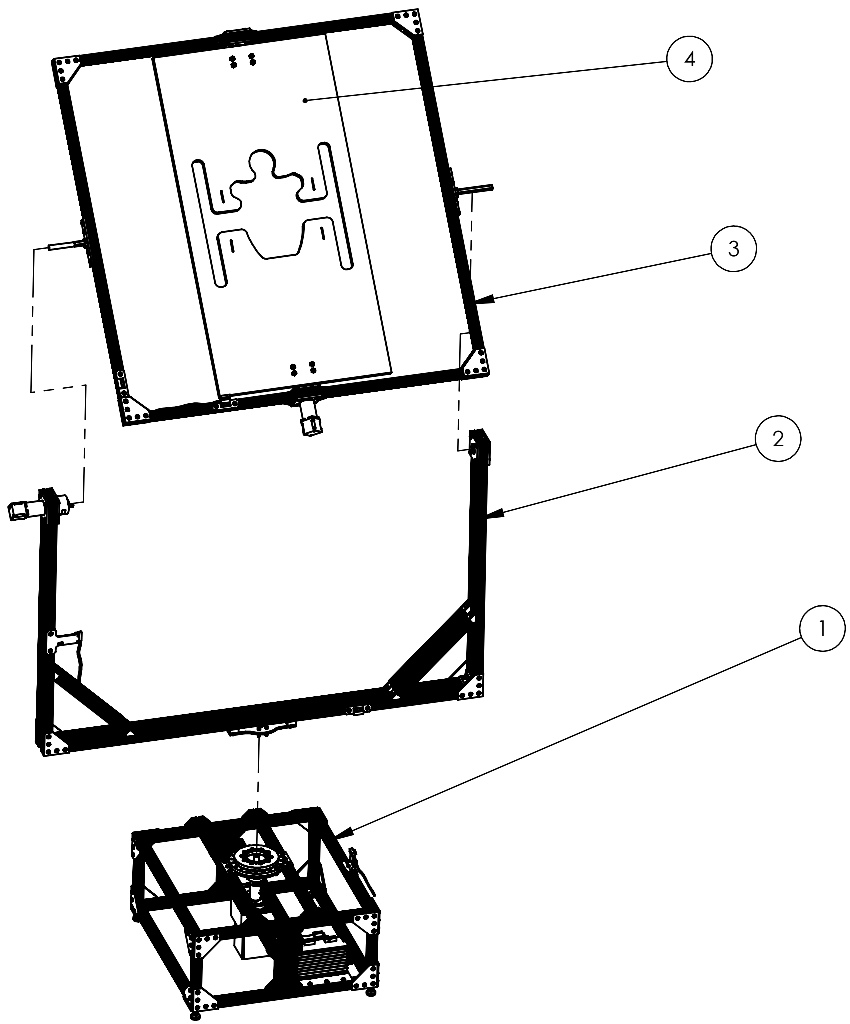
Y	N	
×		1. Will any part of the design create hazardous revolving, reciprocating, running, shearing, punching, pressing, squeezing, drawing, cutting, rolling, mixing or similar action, including pinch points and sheer points?
	×	2. Can any part of the design undergo high accelerations/decelerations?
	×	3. Will the system have any large moving masses or large forces?
	×	4. Will the system produce a projectile?
	×	5. Would it be possible for the system to fall under gravity creating injury?
	×	6. Will a user be exposed to overhanging weights as part of the design?
	×	7. Will the system have any sharp edges?
	×	8. Will any part of the electrical systems not be grounded?
	×	9. Will there be any large batteries or electrical voltage in the system above 40 V?
	×	10. Will there be any stored energy in the system such as batteries, flywheels, hanging weights or pressurized fluids?
	×	11. Will there be any explosive or flammable liquids, gases, or dust fuel as part of the system?
	×	12. Will the user of the design be required to exert any abnormal effort or physical posture during the use of the design?
	×	13. Will there be any materials known to be hazardous to humans involved in either the design or the manufacturing of the design?
	×	14. Can the system generate high levels of noise?
	×	15. Will the device/system be exposed to extreme environmental conditions such as fog, humidity, cold, high temperatures, etc?
	×	16. Is it possible for the system to be used in an unsafe manner?
	×	17. Will there be any other potential hazards not listed above? If yes, please explain on reverse.

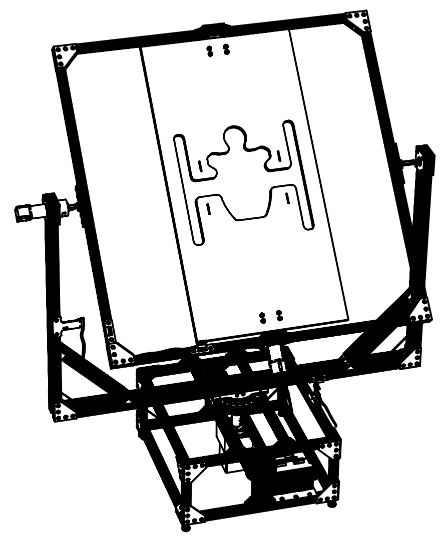
For any "Y" responses, on the reverse side add:
(1) a complete description of the hazard,
(2) the corrective action(s) you plan to take to protect the user, and
(3) a date by which the planned actions will be completed.

Description of Hazard	Planned Corrective Action	Planned Date	Actual Date	Corrective Action
Rotating parts with possible pinch points. Change in rotation due to automatic system control.	Emergency Stop in case of entanglement Post a sign "WARNING: Rotating Parts"	2/6 4/28	1/28	Added ability to shut off all drivers through software in case of emergency. Added an emergency stop in circuit design.

Appendix G: Labeled Exploded View of CAD Model

REVISIONS				
REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	6/5/2020	TVDB





ITEM NO.	ASSEMBLY	QTY.
1	BASE	1
2	FORK	1
3	INNER RING	1
4	DRONE MOUNTING PLATE	1

INSPIRED FLIGHT
SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES

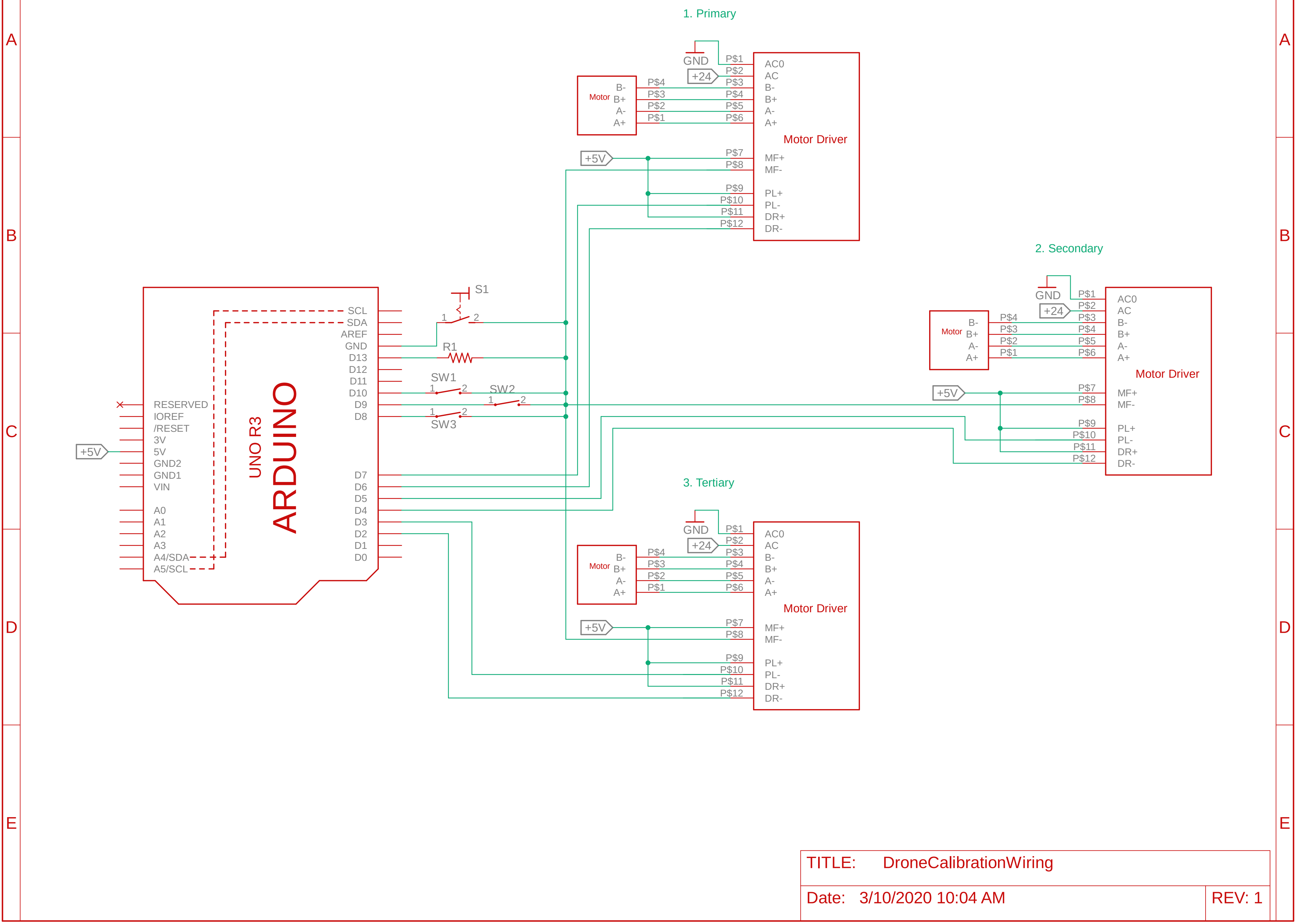
BREAK EDGES	DECIMALS	FINISHED SURFACES
.004/.012	.XX ± .02 .XXX ± .01	.8 Ra
		ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

THIRD ANGLE PROJECTION

DRAWN: TVDB	DATE: 6/5/2020	MAIN ASSEMBLIES EXPLODED	
CHECK/ENG:	DATE:		
APP:	DATE:	SIZE A	PART NO:	REV: A
REFERENCE FILENAME: Final Design Rev_9		SCALE: 1:1	DO NOT SCALE DRAWING	SHEET: 1 OF 1

Appendix H. Wiring Diafram



3/10/2020 10:05 AM f=0.95 D:\HP\EAGLE\projects\DroneCalibrationWiring\DroneCalibrationWiring.sch (Sheet: 1/1)

Wiring Color Chart

This chart shows the which motor driver pin connects with which motor wire. Note: this is consistent for all three motors, but some motors go through wire color changes.

B-	B+	A-	A+
R	B	G	K

This chart traces the color of the wires through solder connections, connectors, and slip rings.

Tertiary Motor											
B	G	R	K								
P	Y	R	K	Tertiary Reed		Secondary Motor					
Br	Y	R	K	G	O	B	G	R	K		
K1	W2	R2	K2	W1	R1	P	Y	R	K	Secondary Reed	
B	W	Pi	DB	Gy	P	Br	Y	R	K	G	O

Appendix I: Software Scripts

DRONE CALIBRATOR SOFTWARE

PYTHON CODE:

```
import serial
import time
import tkinter as tk
from tkinter import Tk, Label, Button, Entry, IntVar, END, W, E
import mavlinkFunctions
from timeit import default_timer as timer

# Initialize variables
state = 0 # MasterMind Task state
aState = 0 # Read Arduino Task state
pState = 0 # Read PixHawk Task state
arduinoState = 0 # Arduino State (read from the Arduino)
PXmsg = '' # Storage for the message from the PixHawk

skip = False # For developers, turns off reed switch locating at the beginning of
each calibration
ardAddress = '/dev/ttyACM0' # Address of the Arduino ('/dev/ttyACM0')
PXaddress = '/dev/ttyUSB0' # Address of the Arduino ('/dev/ttyUSB0')

def spin(stateX,command):
    '''This function is intended to be called in a while loop to make a motor turn.
It sends the turn
    command to the arduino. Then, it makes sure the arduino has changed states
(started turning). Once
    the turn is done (arduinoState = 1) the function returns an incremented
substate, and the MasterMind
    state is incremented. The parameters are the current substate and the command
for which motor to
    turn which way.'''
    # Commands: (+3 = CW Tertiary 90 deg, -3 = CCW Tertiary 90 deg, +2 = CW
Secondary 90 deg
    # -2 = CCW Secondary 90 deg, +1 = spin Primary for ~7s)
    global comReady
    global ard2
    global spinReady
    global state
    if (arduinoState == 1 and comReady): # Arduino in idle state (no motors turning)
        arduino.write(str.encode(str(command))) # send spin command
        comReady = False # Arduino is not ready for another command
    elif (not ard2 and arduinoState != 1): # Arduino goes to state 2 (began
spinning)
        ard2 = True # Spin has started
    elif (arduinoState == 1 and ard2): # Arduino has finished the spin
        comReady = True # Arduino is ready to receive new command
```

```
        ard2 = False # reset ard2 for next spin
        spinReady = False
        if command != 1: # For mag calibration, do not increment state
            state += 1 # position is set, go to corresponding calibrate state
        stateX += 1 # go to next substate
    return stateX

def killMotors():
    arduino.write(str.encode(str(0))) # Turn offPin (arduino) low so no power goes
to motors

def readPX4():
    '''This function is designed to be run in a while loop. While a calibration is
occurring, this
    function reads the global message PXmsg from the PixHawk. It reads the message
and does nothing
    until a critical message comes. If it says to rotate the vehicle, it sets the
spinReady flag to
    True (for mag only). If a side is done calibrating, it sends it back 1 state, to
the corresponding
    orient state. If the whole calibration of a sensor is done, it sends it back a
state and sets the
    done flag to True. Warnings regarding bad orientation will restart the locating
via reed switches.
    Errors restart the singular calibration. [WARN and ERROR messages have not been
tested.]'''
    global state
    global state4
    global state6
    global state10
    global state8
    global done
    global PXmsg
    global spinReady
    global sideDone
    if PXmsg.startswith("INFO  [commander] [cal]"): # All info messages
        if "7 s" in PXmsg: # is in orientation
            spinReady = True
        # Maybe make it calibration done
        elif "calibration done" in PXmsg: # calibration is done
            if (state % 2) == 1: # if in a calibrate (odd number) state
                state -= 1 # go back to orient state
                ln2 = "Calibration has finished"
                app.printMsg(ln1, ln2, ln3)
            done = True # Flag to carry on to next sensor calibration
        elif "side done" in PXmsg or (state == 9 and sideDone): # if a side is done
calibrating
            if (state % 2) == 1 and not spinReady: # if in a calibrate (odd number)
state
                state -= 1 # go back to orient state
```

```
                sideDone = False # Reset flag for next side
            else:
                sideDone = True # Side is calibrated
    elif PXmsg.startswith("WARN"):
        if "invalid orientation" in PXmsg:
            # Restart arduino calibration
            ln2 = "Error: Need to restart whole thing. Invalid Orientation"
            app.printMsg(ln1, ln2, ln3)
            state = 0
        else:
            ln2 = "Please wait" # It's ok. Wait it out.
            app.printMsg(ln1, ln2, ln3)
    elif PXmsg.startswith("ERROR"):
        # Restart whichever calibration it is on
        ln2 = "Error: Caibration Failed. Please restart"
        app.printMsg(ln1, ln2, ln3)
        state = int(state/2)*2 # Return to corresponding orient state (even
numbered)
        state4 = 0 # Reset corresponding substate
        state6 = 0
        state10 = 0
        state8 = 0

def calibrate(sensor):
    ''' This function takes the sensor requiring calibration and starts the
calibration.
    Gyroscope = 'gyro', Level Horizon = 'level', Accelerometer = 'accel', Compass =
'mag'. '''
    mavlinkobj.write("commander calibrate " + sensor + "\n") # write to mavlink
object

class Application(tk.Frame):
    ''' This class is the GUI. It creates an app window with buttons and a readout
for the user.'''
    def __init__(self, master=None):
        super().__init__(master)
        self.master = master
        master.title("Drone Calibrator")
        master.geometry("")
        master.configure(bg='blue')

        # Text
        self.display = " \nWelcome to the amazing new drone calibrator!
\nLoading..."
        self.total_label_text = IntVar()
        self.total_label_text.set(self.display)
        self.total_label = Label(master,
textvariable=self.total_label_text,bg='blue',fg='white')
        self.label = Label(master, text = "   Drone
Calibrator",bg='blue',fg='white')
```

```
        self.total_label_text.set(self.display)

        # Buttons
        self.start_button = Button(master, text="Full Calibration", command=lambda:
self.start(),bg='purple', fg = 'white')
        self.quit_button = Button(master, text="Quit",
command=self.quit,bg='purple', fg = 'white')
        self.gyro_button = Button(master, text="Gyroscope", command=lambda:
self.gyro(),bg='green',fg = 'white')
        self.level_button = Button(master, text="Level Horizon", command=lambda:
self.level(),bg='green',fg = 'white')
        self.accel_button = Button(master, text="Accelerometer", command=lambda:
self.accel(),bg='green',fg = 'white')
        self.mag_button = Button(master, text="Compass", command=lambda:
self.mag(),bg='green',fg = 'white')

        # LAYOUT
        self.label.grid(row=0, column=0, columnspan=4, sticky=W, pady=10)
        self.total_label.grid(row=1, column=0, columnspan=4, sticky=W+E)

        self.start_button.grid(row=2, column=0, columnspan=2, pady=7)
        self.quit_button.grid(row=2, column=3, pady=7)
        self.gyro_button.grid(row=3, column=0, padx=5, pady=5)
        self.level_button.grid(row=3, column=1, padx=5, pady=5)
        self.accel_button.grid(row=3, column=2, padx=5, pady=5)
        self.mag_button.grid(row=3, column=3, padx=5, pady=5)

    def start(self): # Full Calibration button
        global state
        global indCalib
        if state == 1:
            state = 2
            indCalib = 0

    def printMsg(self, ln1, ln2, ln3):
        self.display = ln1 + "\n" + ln2 + "\n" + ln3
        self.total_label_text.set(self.display)
        self.update()

    def quit(self): # Quit button
        killMotors() # Turn off motors
        self.master.destroy() # Close the app

    def gyro(self): # Gyroscope button
        global state
        if state == 1:
            global state4
            global state5
            global indCalib
            indCalib = 4 # orient state of the gyro calibration
```

```
            state = 2
            state4 = 0
            state5 = 0
    def level(self): # Level Horizon button
        global state
        if state == 1:
            global state6
            global state7
            global indCalib
            indCalib = 6 # orient state of the level calibration
            state = 2
            state6 = 0
            state7 = 0
    def accel(self): # Accelerometer button
        global state
        if state == 1:
            global state10
            global state11
            global indCalib
            indCalib = 10 # orient state of the accel calibration
            state = 2
            state10 = 0
            state11 = 0
    def mag(self): # Compass button
        global state
        if state == 1:
            global state8
            global state9
            global indCalib
            indCalib = 8 # orient state of the mag calibration
            state = 2
            state8 = 0
            state9 = 0

# Start the app
root = tk.Tk()
app = Application(master=root)

while True:
    # MasterMind Task
    readPX4()
    while True:
        # 0 Initialization
        if state == 0:
            # Initialize all variables
            state1 = 0 # substates
            state2 = 0
            state3 = 0
            state4 = 0
            state5 = 0
```

```
            state6 = 0
            state7 = 0
            state8 = 0
            state9 = 0
            state10 = 0
            state11 = 0
            state12 = 0
            if arduinoState == 1: # Waits for arduino connection to go to state 1
                state += 1
            indCalib = 0 # Individual calibration flag, full calibration = 0, gyro =
4, level = 6,
            # accel = 8, mag = 10
            ln1 = 'Inspired Flight'# Default messages for GUI
            ln2 = 'Load and power on drone'
            ln3 = 'Ready to Calibrate'
            ard2 = False # Flag used in spin function for if arduino is out of state
1 (spin started)
            comReady = True # Flag used in spin function for if the arduino is
accepting communication
            done = False # Flag used in readPX4 function to tell if sensor
calibration is done
            spinReady = False # Flag used in readPX4 function, True while Primary
spins in mag calibration
            sideDone = False # Flag used in readPX4 function to indicate a side is
calibrated
            break

        # 1 Wait for Start
        elif state == 1:
            app.printMsg(ln1, ln2, ln3)
            # Button press increments state
            break

        # 2 Connect to Drone via Telemetry
        elif state == 2:
            if state2 == 0:
                if not PXmsg: # if not already connected, try to connect
                    ln2 = "Connecting to drone"
                    app.printMsg(ln1, ln2, ln3)
                    pState = 1 # move Read PixHawk task to communication state
                    mavlinkobj =
mavlinkFunctions.MavlinkSerialPort(PXaddress,57600,timer(),devnum=10)
                    mavlinkobj.write('\n') # make sure the shell is started
                    state2 += 1
            if PXmsg: # Check if connected
                ln2 = "Drone connected"
                app.printMsg(ln1, ln2, ln3)
                state += 1
            break
```

```
# 3 Locate
elif state == 3:
    if state3 == 0 and not skip:
        ln2 = "Orienting Calibrator"
        app.printMsg(ln1, ln2, ln3)
        state3 = spin(state3,9) # Start locating with reed switches
        state = 3 # spin function moves it to 4 too early
    else: # Once located and in starting position
        if indCalib != 0: # Go to individual calibration
            state = indCalib
        else: # Full Calibration, go to next calibration
            state += 1
    break

# 4 Gyroscope - Orient
elif state == 4:
    if state4 == 0:
        ln1 = "Gyroscope Calibration"
        ln2 = "Starting Calibration"
        app.printMsg(ln1, ln2, ln3)
        # Do nothing (no turns)
        state += 1
        state4 += 1
    else:
        readPX4() # wait for calibration to finish
        if done: # Calibration is done
            if indCalib == 0: # Full Calibration, go to next calibration
                state = 6
                PXmsg = '' # clear message from last calibration
            else: # Return to beginning to await button press
                state = 1
            ln2 = "Gyroscope Calibration Complete"
            app.printMsg(ln1, ln2, ln3)
            sideDone = False # clear flags
            spinReady = False
            done = False
    break

# 5 Gyroscope - Hold
elif state == 5:
    if state5 == 0:
        state5 += 1
        calibrate("gyro") # start calibration
        ln2 = "Calibrating"
        app.printMsg(ln1, ln2, ln3)
    readPX4() # wait for calibration to finish, then return to orient state
    break

# 6 Level Horizon - Orient
elif state == 6:
```

```
            # print(done)
            if state6 == 0:
                ln1 = "Level Horizon Calibration"
                ln2 = "Starting Calibration"
                app.printMsg(ln1, ln2, ln3)
                # Do nothing
                state += 1
                state6 += 1
            else:
                readPX4() # wait for calibration to finish
                if done:
                    if indCalib == 0: # Full Calibration, go to next calibration
                        state = 8
                        PXmsg = '' # clear message
                    else: # Return to beginning to await button press
                        state = 1
                    ln2 = "Level Horizon Calibration Complete"
                    app.printMsg(ln1, ln2, ln3)
                    sideDone = False # clear flags
                    spinReady = False
                    done = False
            break

        # 7 Level Horizon - Hold
        elif state == 7:
            if state7 == 0:
                state7 += 1
                calibrate("level") # start calibration
                ln2 = "Calibrating"
                app.printMsg(ln1, ln2, ln3)
            readPX4() # wait for calibration to finish, then return to orient state
            break

        # 8 Compass - Orient
        elif state == 8:
            turns = [3,2,2,-3,2,2] # the order of turns within calibration
            # (+3 = CW Tertiary 90 deg, -3 = CCW Tertiary 90 deg, +2 = CW Secondary
90 deg
            # -2 = CCW Secondary 90 deg, +1 = spin Primary for ~7s)
            if state8 == 0:
                if arduinoState == 1:
                    ln1 = "Compass Calibration"
                    ln2 = "Starting Calibration"
                    app.printMsg(ln1, ln2, ln3)
                    state8 = 1
                    state = 9 # position is set, go to calibrate

            elif state8 >= 1 and state8 <= 5:
                ln2 = "Calibrating"
                app.printMsg(ln1, ln2, ln3)
```

```
                state8 = spin(state8,turns[state8-1]) # turn tertiary motor 90 deg
CW, then go to next state and substate
            elif state8 == 6:
                state8 = spin(state8,turns[state8-1])
                state = 8 # Calibration is done, don't go to state 9
                readPX4() # wait for calibration to finish
            else:
                readPX4() # wait for calibration to finish
                if done:
                    if indCalib == 0: # Full Calibration, go to next calibration
                        state = 10
                        PXmsg = '' # clear message
                    else: # Return to beginning to await button press
                        state = 1
                    ln2 = "Compass Calibration Complete"
                    app.printMsg(ln1, ln2, ln3)
                    sideDone = False # clear flags
                    spinReady = False
                    done = False
            break

        # 9 Compass - Spin
        elif state == 9:
            if state9 == 0:
                state9 += 1
                calibrate("mag") # start calibration
            elif spinReady: # once PixHawk is ready to be spun
                spin(1,1) # turn Primary axis for ~7s
            readPX4() # wait for calibration and spin to finish, then return to
orient state
            break

        # 10 Accelerometer - Orient
        elif state == 10:
            turns = [3,2,2,-3,2,2] # the order of turns within calibration
            # (+3 = CW Tertiary 90deg, -3 = CCW Tertiary 90 deg, +2 = CW Secondary
90 deg
            # -2 = CCW Secondary 90 deg, +1 = spin Primary for ~7s)
            if state10 == 0:
                if arduinoState == 1:
                    ln1 = "Accelerometer Calibration"
                    ln2 = "Starting Calibration"
                    app.printMsg(ln1, ln2, ln3)
                    state10 = 1
                    state = 11 # position is set, go to calibrate
                    comReady = True
            elif state10 >= 1 and state10 <= 5:
                ln2 = "Calibrating"
                app.printMsg(ln1, ln2, ln3)
                state10 = spin(state10,turns[state10-1]) # turn tertiary motor 90
```

```
deg CW, then go to next state and substate
            elif state10 == 6:
                state10 = spin(state10,turns[state10-1])
                state = 10 # Calibration is done, don't go to state 11
                readPX4()
            else:
                readPX4() # wait for calibration to finish
                if done:
                    if indCalib == 0: # Full Calibration, go to next calibration
                        state = 12
                        PXmsg = '' # clear message
                    else: # Return to beginning to await button press
                        state = 1
                    ln2 = "Accelerometer Calibration Complete"
                    app.printMsg(ln1, ln2, ln3)
                    sideDone = False # clear flags
                    spinReady = False
                    done = False
            break

        # 11 Accelerometer - Hold
        elif state == 11:
            if state11 == 0:
                state11 += 1
                calibrate("accel") # start calibration
            readPX4() # wait for calibration to finish, then return to orient state
            break

        # 12 End
        elif state == 12:
            # Should be oriented in unloading position
            ln1 = "Calibration Complete. Please remove drone'"
            ln2 = "Press 'Quit' to release motors"
            app.printMsg(ln1, ln2, ln3)
            time.sleep(5) # Time to read messages
            state = 0
            break
        break # in case it didn't already break (shouldn't be used)

    # Read Arduino Task
    while True:
        # 0 Initialization
        if aState == 0:
            aState = 1
            arduino = serial.Serial(ardAddress,9600) # Connect to arduino via serial
        elif aState == 1:
            time.sleep(2) # requires delay to connect before taking commands
            arduino.write(str(10).encode()) # Send any command to get arduino out of
state 0
            aState += 1
```

```
        # 1 Read Arduino
        elif aState == 2:
            if arduino.in_waiting: # if there is a message
                arduinoState = ord(arduino.read()) # read the arduino message
                # print('arduino state: ' + str(arduinoState)) # For developers,
print arduino state in terminal
        break

    # Read PixHawk Task
    while True:
        # 1 Read PixHawk
        if pState == 1: # once connection is made
            mavlinkobj.heartbeatFun(timer()) # send heartbeat
            data = mavlinkobj.read(4096) # read the PixHawk
            if data and len(data) > 0: # if there is a message
                PXmsg = data # save the message
                # print(PXmsg) # For developers to read all messages
                if PXmsg.startswith("INFO  [commander] [cal]"): # All info messages
                    PXindex = 23
                    ln3 = PXmsg[PXindex:-1] # print message without heading
                    app.printMsg(ln1, ln2, ln3)
                elif PXmsg.startswith("WARN"): # Warnings
                    PXindex = 17
                    ln3 = PXmsg[PXindex:]
                    app.printMsg(ln1, ln2, ln3)
                elif PXmsg.startswith("ERROR"):
                    PXindex = 17
                    ln3 = PXmsg[PXindex:]
                    app.printMsg(ln1, ln2, ln3)
        break
    # print(state) # For developers to show MasterMind state

    # Update App Task
    app.update()
```

ARDUINO CODE:

```
int stepPin = 3;        // Step pin, on falling edge
int dirPin = 2;         // changes direction
int offPin = 13;        // Low = motors unpowered
long step90 = 10000;    // step 90 deg (40000 steps = 1 rev)
long step7 = 7000;      // steps to last 7s
long steps = step90;    // variable, total steps to take
int Dlay = 400;         // Time between steps (microseconds)
int rampTime = 250;     // Number of increments in increasing step speed for ramp
long microsCompare = 0; // Counter to increase rate in ramp (ONLY GOOD FOR 70 min)
long stepCount = 0;     // Counter for steps taken
int command;            // Serial read byte (1=1spin, -1=1stop spin, 2=CW2, -2=CCW2,
3=CW3, -3=CCW3) (+/-2 and +/-3 are 90 deg and +/-1 spins until stop)
int state = 0;

//Variables for motor homing
int reedPin = 0;            // Pin for the current reed sensor
int pinReedPrimary = 10;    // pins for Reed switch wiring
int pinReedSecondary = 9;
int pinReedTertiary = 8;
int sensorOffset = 0;       // number of steps to take after sensor has been
triggered
int sensorOffset1 = 2800;   // value which may reuqire calibration, for primary
motor
int sensorOffset2 = 9500;
int sensorOffset3 = 1850;
int sensorval = 0;          // Value of the current read sensor
int initCommand = 1;        // Command to determine which axis should be running
int noiseThreshold;         // Variable to account for various amounts of sensor
noise
int sensorNum;              // Current number of consecutive sensor trigger values
int sensorTriggered = 0;    // Variable set once the sensor has been triggered
int initComplete = 0;       // Variable set when the homing sequence is complete
int firstrun;               // Set so the motor and reed pins can be changed to the
new axis

void setup(){
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
    while(!Serial){
    }
  pinMode(0,INPUT);

  // set up pins for output
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
```

```
  pinMode(6,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(offPin,OUTPUT);

  // Set up reed switch pins for input
  pinMode(pinReedPrimary,INPUT_PULLUP);
  pinMode(pinReedSecondary,INPUT_PULLUP);
  pinMode(pinReedTertiary,INPUT_PULLUP);
  // Initialize the motor command pins
  digitalWrite(offPin,HIGH);
  digitalWrite(stepPin,HIGH);
}
void loop(){

  if (Serial.available()) {
    // read the incoming byte from python script:
    command = Serial.readString().toInt(); // reads command from computer (converted
from ASCII)

    if (command==0 || command==8) { // turn motor on (8) or off (0)
      if (command==0) {
        digitalWrite(offPin, LOW); // turn off all motors
      }
      else {
        digitalWrite(offPin, HIGH); // turn on all motors
      }
        command = 11;
  }
  }
switch (state) {
  case 0:
    if (command==10){
      // once python script has started, enter the idle state
      state = 1;
      Serial.write(state);
      command = 11;
    }
    break;

// Idle Wait For Command
  case 1:
     // Choose Motor
     if (command==3 || command==-3) {//Tertiary
      stepPin = 3; // define pins for appropriate motor
      dirPin = 2;
      steps = step90; // set steps to turn 90 degrees
      Dlay = 500;
     }
     else if (command==2 || command==-2) {//Secondary
      stepPin = 5;
```

```
      dirPin = 4;
      steps = step90; // set steps to turn 90 degrees
      Dlay = 500;
     }
     else if (command == 1) {//Primary
      stepPin = 7;
      dirPin = 6;
      steps = step7; // set steps to about 7 seconds.
      Dlay = 1000;
     }

    // Set Direction Pin
    if (command==1 || command==2 || command==3) {
      digitalWrite(dirPin,LOW); // set to CW
    }
    else if (command==-2 || command==-3) {
      digitalWrite(dirPin, HIGH); // set to CCW
    }

    // if go command
    if (command==1 || command==2 || command==3 || command==-2 || command==-3) {
      state = 2;
      Serial.write(state);
      microsCompare = micros();
      stepCount = 0;
    }
    if (command == 9){// Error occured, go back to homing
      state = 5;
      initCommand = 1;
      firstrun = 1;
      initComplete = 0;
      Serial.write(state);
    }
    if (command!=11) {
      command = 11;
    }

  break;

// Ramp to Speed
  case 2:
     if (micros() >= microsCompare && stepCount < rampTime) {
         // Step
       digitalWrite(stepPin,LOW);
       digitalWrite(stepPin,HIGH);
       stepCount++;
       microsCompare += sqrt(rampTime/stepCount)*Dlay; // slowly decrease delay
between steps
      }
      else if (stepCount >= rampTime) {
```

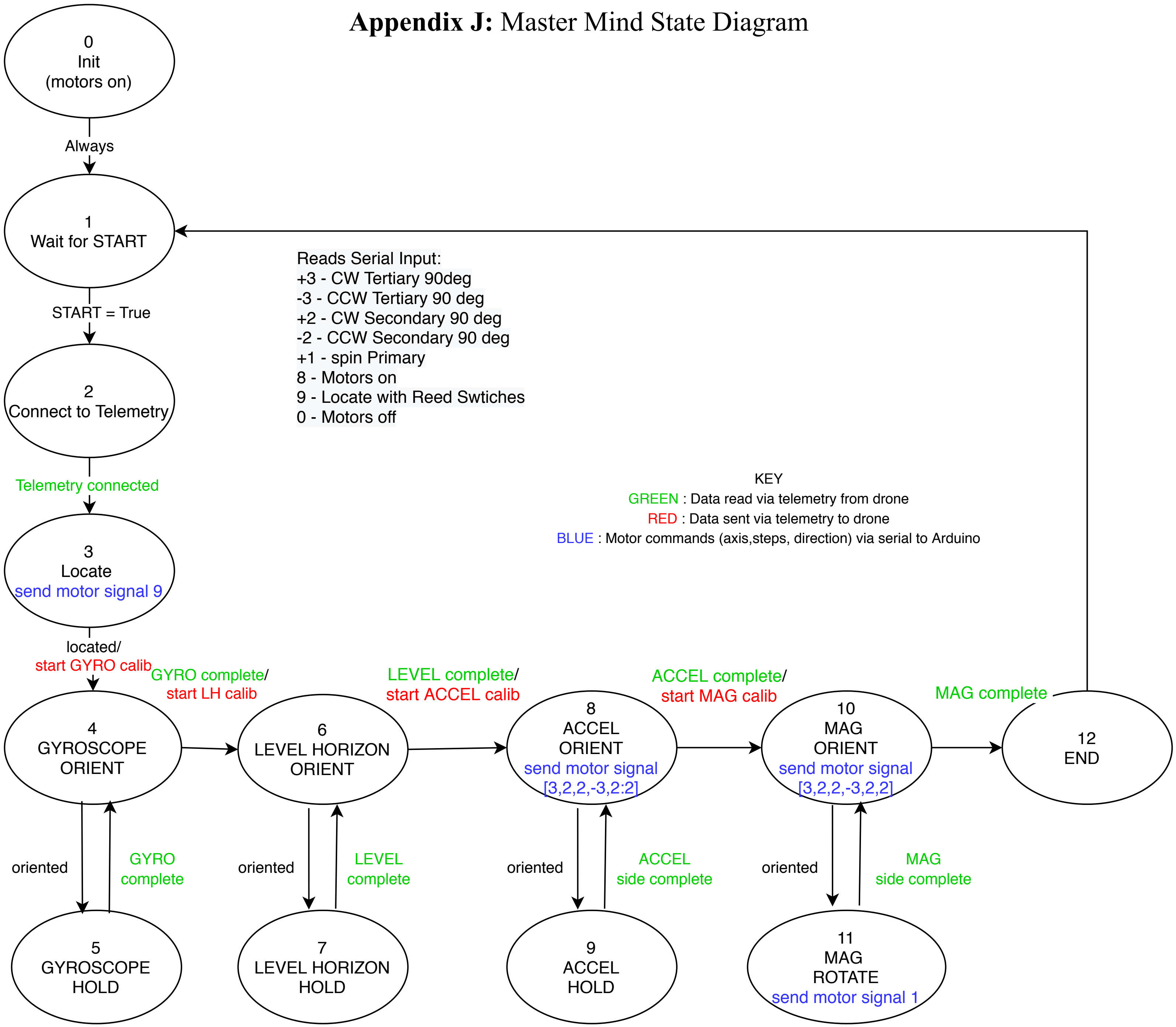
```
        state = 3;
        Serial.write(state);
      }
      break;

// Turn at Constant Speed
  case 3:
    if (micros() >= microsCompare && (stepCount < (steps - rampTime) || command ==
1)) {
      //if more steps to go or primary still says spin
        // Step
      digitalWrite(stepPin,LOW);
      digitalWrite(stepPin,HIGH);
      stepCount++;
      microsCompare += Dlay;
    }
    else if ((stepCount >= (steps - rampTime) || command == -1)) {
      //if no more steps to go or primary says stop spin
      state = 4;
      Serial.write(state);
      if (command == -1) {
        steps = stepCount+rampTime; // fake the amount of steps required to allow
ramp down
      }
    }
    break;

 // Ramp to Stop
   case 4:
     if (micros() >= microsCompare && stepCount < steps) {
         // Step
       digitalWrite(stepPin,LOW);
       digitalWrite(stepPin,HIGH);
       stepCount++;
       microsCompare += sqrt(rampTime/(steps - stepCount))*Dlay; // slowly decrease
delay between steps
      }
      else if (stepCount >= steps) {
        state = 1;
        Serial.write(state);
     }
   break;

   case 5:
   //Homing sequence for all three motors
        if (firstrun==1){
          // Choose Motor
          if (initCommand==3) {//Tertiary axis
            stepPin = 3; //define pins for appropriate motor
            dirPin = 2;
```

```
    steps = 500000; //set steps to turn over a full rotation
    reedPin = pinReedTertiary;
    sensorOffset = sensorOffset3;  //offset steps for Tertiary motor
    //resetting values for new axis
    firstrun = 0;
    stepCount = 0;
    sensorNum = 0;
    noiseThreshold = 10;
    Dlay = 400;
    rampTime = 400;
    microsCompare = micros();
  }
  else if (initCommand==2) {//Secondary
    stepPin = 5;  //define pins for appropriate motor
    dirPin = 4;
    steps = 50000;  //set steps to turn over a full rotation
    reedPin = pinReedSecondary;
    sensorOffset = sensorOffset2;  //offset steps for Secondary motor
    //resetting values for new axis
    firstrun = 0;
    stepCount = 0;
    sensorNum = 0;
    noiseThreshold = 7;
    Dlay = 400;
    digitalWrite(dirPin, HIGH);
    rampTime = 250;
    microsCompare = micros();
  }
  else if (initCommand == 1) {//Primary
    stepPin = 7;  //define pins for appropriate motor
    dirPin = 6;
    steps = 2000000000;   //set steps to turn over a full rotation
    reedPin = pinReedPrimary;
    sensorOffset = sensorOffset1;  //offset steps for Primary motor
    //resetting values for new axis
    firstrun = 0;
    stepCount = 0;
    sensorNum = 0;
    noiseThreshold = 3;
    Dlay = 2000;
    rampTime = 1000;
    microsCompare = micros();
  }
}

//Ramp down for initialization motors
if (micros() >= microsCompare && (stepCount >= (steps - rampTime))) {
  // Step
  digitalWrite(stepPin,LOW);
  digitalWrite(stepPin,HIGH);
```

```
          stepCount++;
          // slowly increase delay between steps
          microsCompare += sqrt(rampTime/(steps - stepCount + 1))*Dlay;
        }
        else if (stepCount >= steps) {
          //Reached step value so axis is complete
          sensorTriggered = 0;
          initCommand += 1;
          firstrun = 1;
          if (initCommand == 4){ //once all three axis have been complete, the
homing process is done
            initComplete = 1;
            state = 1;
            Serial.write(state);
          }
        }

          //Ramp up for initalization motors
          else if (micros() >= microsCompare && (stepCount < rampTime)) {
            // Step
            digitalWrite(stepPin,LOW);
            digitalWrite(stepPin,HIGH);
            stepCount++;
            //Test to see if sensor has already been triggered
            if (sensorTriggered==0){
              sensorval = digitalRead(reedPin);
              if (sensorval == 0){ // reed sensor triggered
                //increment the SensorNum so noise doesn't trigger sensor
                sensorNum += 1;
                if (sensorNum >= noiseThreshold){
                  //once past the noise threshold, it is a valid trigger
                  steps = stepCount + sensorOffset;
                  sensorTriggered = 1;
                }
              }
              else if (sensorval == 1){
                //if sensor is not triggered, reset any possible noise signals
                sensorNum = 0;
              }
            }
            // slowly decrease delay between steps
            microsCompare += sqrt(rampTime/stepCount)*Dlay;
          }

          //Constant steps for initalization
          else if (micros() >= microsCompare && (stepCount <= (steps-rampTime))) {
            // Step
            digitalWrite(stepPin,LOW);
            digitalWrite(stepPin,HIGH);
            stepCount++;
```

```
          //Test to see if sensor has already been triggered
          if (sensorTriggered==0){
            sensorval = digitalRead(reedPin);
            if (sensorval == 0){ // reed sensor triggered
              //increment the SensorNum so noise doesn't trigger sensor
              sensorNum += 1;
              if (sensorNum >= noiseThreshold){
                //once past the noise threshold, it is a valid trigger
                steps = stepCount + sensorOffset;
                sensorTriggered = 1;
              }
            }
            else if (sensorval == 1){
              //if sensor is not triggered, reset any possible noise signals
              sensorNum = 0;
            }
          }
          //set time between next step
          microsCompare += Dlay;
        }

  if (initComplete==1){
     //when all axis complete, go to idle state
     state = 1;
     Serial.write(state);
  }
  break;

}
}
```

Appendix J: Master Mind State Diagram



INSPIRED FLIGHT

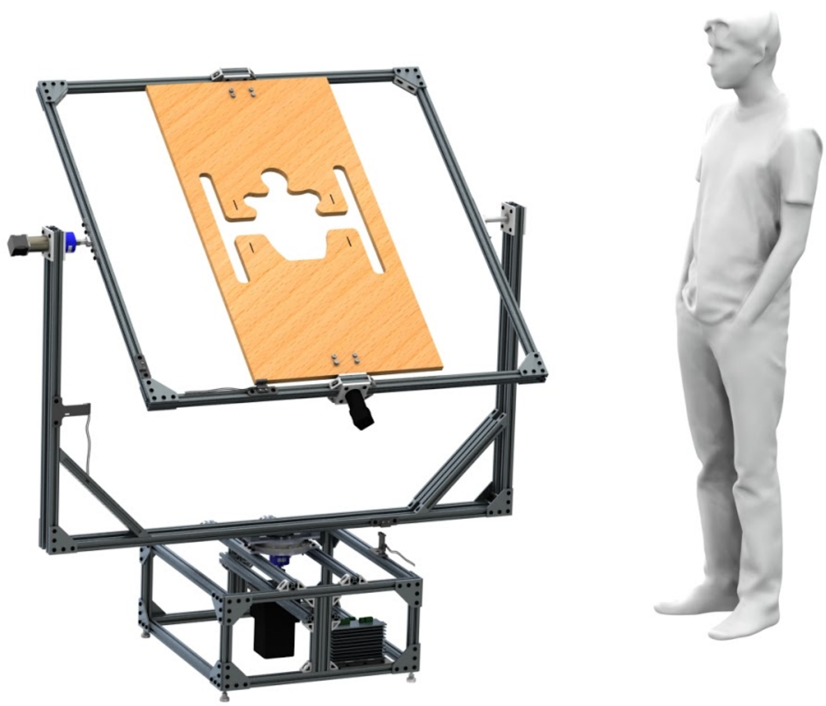
DRONE CALIBRATION SYSTEM

USER MANUAL V 1.0

CAL POLY SENIOR PROJECT 2020

Appendix K

Table of Contents



1.Safety Information

The calibration machine has several high powered components. Do not touch any electrical wiring unless both the arduino and the motor drivers are unpowered. It is also recommended that the operator places the emergency stop near their workstation so they can quickly shut off the machine in the event of an error. Due to the moving components of the machine, please stay 5 ft away from any moving parts during operation.

2.Product Description & Specifications

SYSTEM DESCRIPTION

This is a calibration system designed for Inspired Flight to automate the previously manual process of calibrating their IF750 quadcopter. The design is similar to a gyroscope, and consists of 3 axes of rotation. Motors that spin the mounted drone about these axes orient in any position, allowing for a hands-free calibration.

SYSTEM SPECIFICATIONS

Primary Motor - Nema 34, 12,700 oz-in, 20:1 Spur Gearbox
Secondary/Tertiary Motors - Nema 17, 5500oz-in, 100:1 Planetary Gearbox
Power Supply - 24V, 10A
Primary Motor Driver - 2.4-7.2A, AC 18V-80V, DC 36V-110V
Secondary/Tertiary Motor Drivers - 1.0-4.2A, DC 20V-50V

AIRCRAFT DESCRIPTION

The system was designed for use with an Inspired Flight IF750 Quadcopter. Motors, drivers, and other physical aspects of the system were driven from the size and weight of this quadcopter. Similarly sized aircrafts/revisions of the IF750 can be implemented by replacing the drone mounting plate with a new cutout for the aircraft.

3.System Overview

The calibration system consists of four main subassemblies: the base, fork, inner ring, and drone mounting plate. These are labeled in Figure 1.

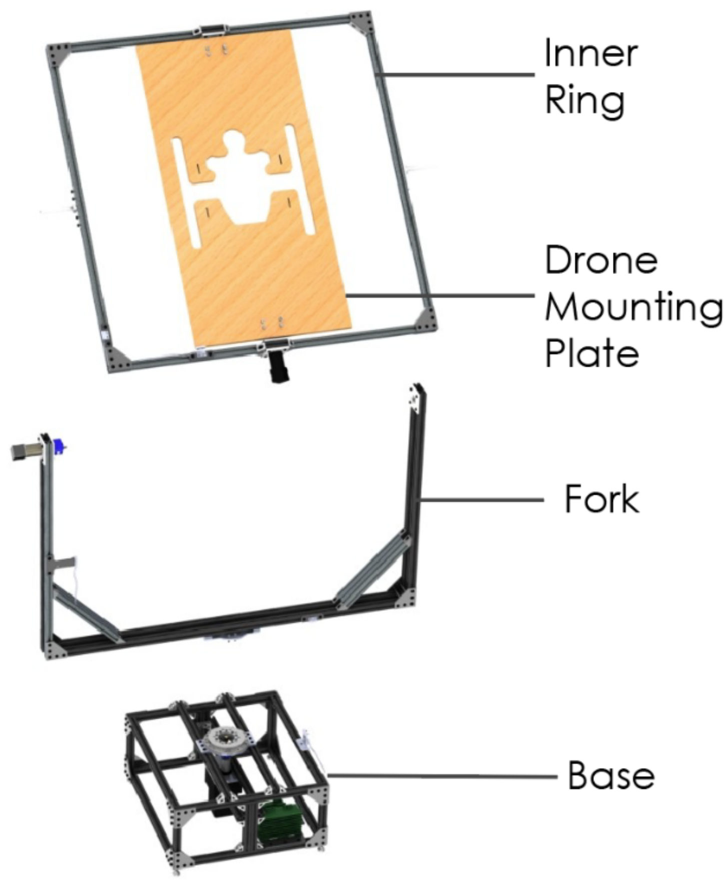


Figure 1: Exploded Diagram of Main Subassemblies

The system is similar to a gyroscope, and has 3 axes of rotation, seen in Figure 2. Each of these axes are driven by a single stepper motor, and allow the mounted drone to be oriented and spun in any direction. The system is programmed to communicate with the drone's flight controller using wireless telemetry, allowing for hands-free, consistent calibration.

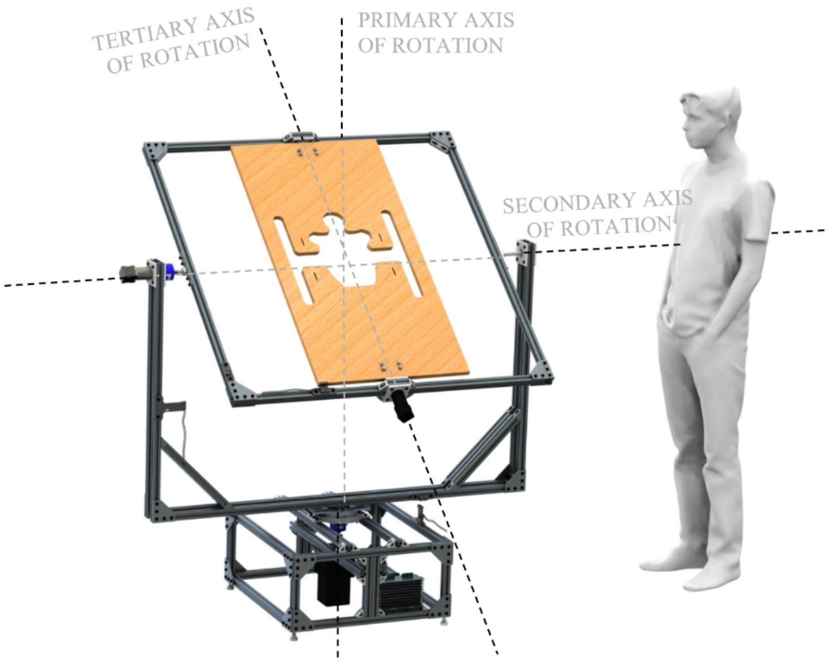


Figure 2. System's Rotational Axes

Apart from the system itself, the setup must also include a table or stand for the Linux computer and emergency stop button, as seen in Figure 3.

Figure 3: Overall Setup of Calibration System

4.Software Overview

The software which we designed is in two parts. The first is written in Python and utilizes a python MAVlink library. This will run on the controlling computer and will interface with the PixHawk (the drone module being calibrated), run the Master Mind code, and communicate over serial with the Arduino Uno. The second portion of the software is the C++ code running on the Arduino Uno. This code interprets motor commands from the controlling computer and sends them to the motor drivers. The organization of the three computers running the software, the Linux computer, the Arduino, and the PixHawk, are shown in Figure 4.

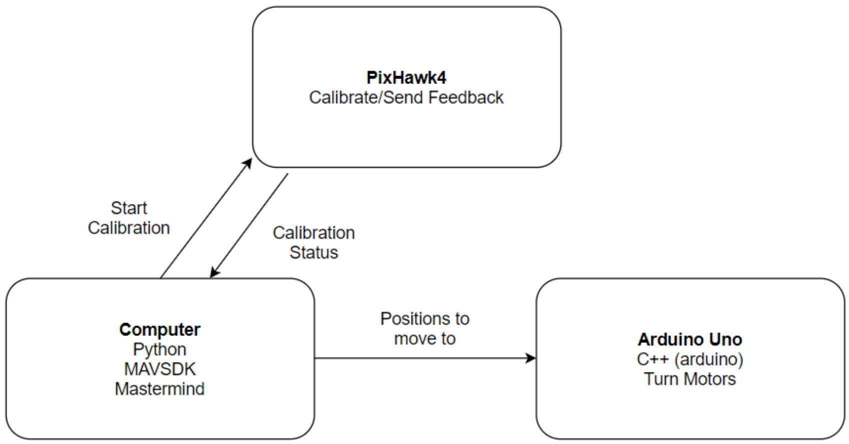


Figure 4. Communication Diagram of Three Main Computers

Arduino

The Arduino code is broken into five states. The states are shown below in Figure 5. In the first state, each of the three axes is run until it detects a reed switch, then it rotates to the starting position for calibration.

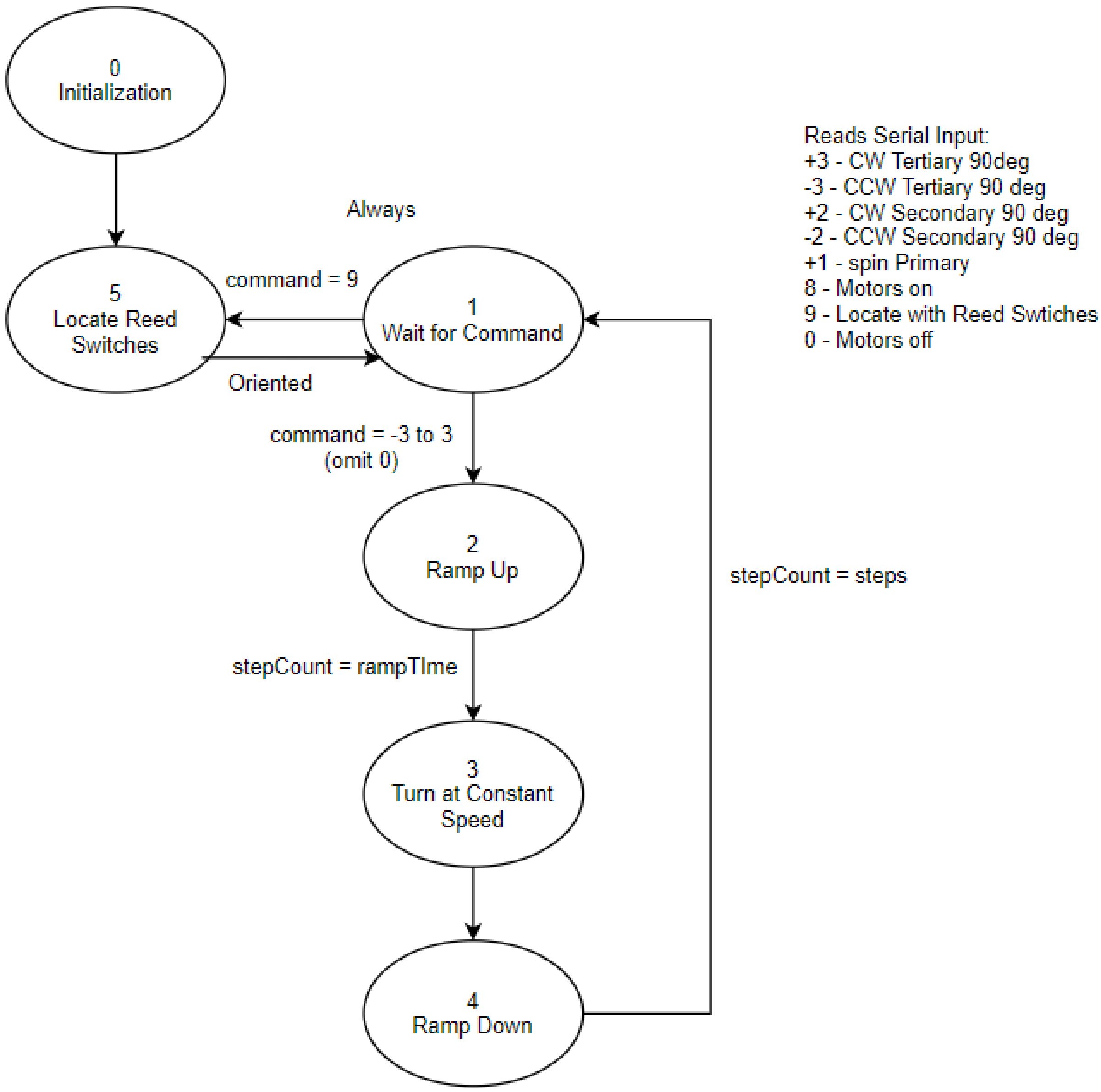


Figure 5. Arduino State Diagram

In state one, it waits for a serial command from the computer. 1 will spin the primary axis for about seven seconds. 2 or -2 will spin the secondary axis in either direction 90° and 3 or -3 will turn the tertiary axis. To turn the motors on and off, the commands are 8 and 0 respectively. For each of the spin commands, (-3, -2, 1, 2, 3), the code advances to state two ramp up to speed, state three to turn the appropriate amount, and then state four to ramp to a stop, returning to state one to await another command. Upon entering each state, the Arduino sends its current state to the Master Mind python code.

State five runs the motors one at a time until the reed switch on each axis is triggered. Once triggered, the motor will orient the axis to its starting position. This orientation state can be accessed by sending the Arduino a spin command of 9. This allows for the machine to be re-oriented between calibrations.

Every clock cycle, the Arduino checks for a 8 or 0 to turn on or off the motors, but all other commands are only processed if the state is 1. This allows for a 0, or stop command, to be received and interpreted at any time.

Python

The code that runs on the control computer is written in python and has only been tested on a linux system. The python script creates a graphical user interface (GUI) using the Tkinter library and is shown in Figure 6. The GUI displays several buttons that allow the user to run either the full calibration or calibrate a specific sensor. There are three tasks that run on the python script. One task handles the communications with the Arduino over a serial connection. The second task communicates with the PixHawk through serial communication. By sending commands directly to the Nuttx shell running on the PixHawk, this task can initiate the calibration process on the drone. The python script is also able to read the feedback from the Nuttx shell to determine if the system is ready for the next step in the calibration process. The third task, Master Mind is the most complex and we will be discussing it in depth.

Figure 6. GUI generated by the python script

Figure 7 shows the state diagram of Master Mind. Master Mind runs through a startup sequence and then calibrates four sensors. The “Orient” state will signal the motors to move the drone to each of six orientations. The “Hold” state will be a delay for the PixHawk Module to calibrate. The “Rotate” state will rotate the base while the PixHawk calibrates. Once a sensor is calibrated, Master Mind will move on to the next sensor calibration.

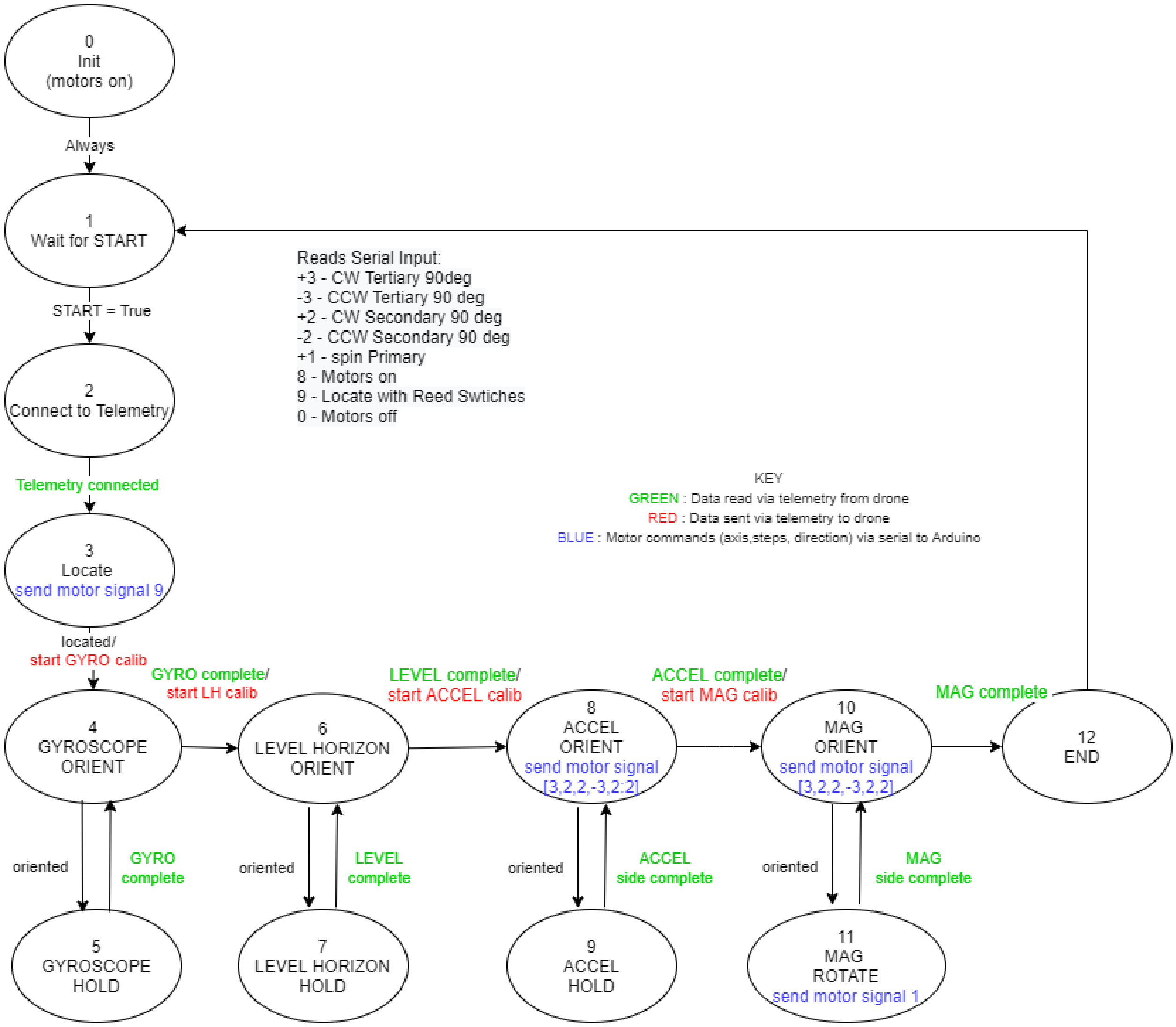


Figure 7. Final Master Mind State Diagram

Detailed Overview of Master Mind

Each orient state has substates corresponding to the different orientations the system must reach before completing calibration. The messages displayed in the GUI that correspond to the calibration are set in the initialization substate. Master Mind then transitions to the hold state, where it sends the appropriate message to start the sensor calibration on the PixHawk. Once the calibration has been started, Master Mind continues to read the messages from the PixHawk until it receives a confirmation that a sensor is done calibrating, or a side is done. It will then transition back to the orient state from there either increment substates or move on to the next sensor. This repeats for each of the four sensors. The states and substates are largely controlled by the spin and readPX4 function.

There are several functions created for the purpose of switching states and sending serial commands. The spin function sends a command to the Arduino to initiate a movement on the calibration machine. This spin function has some additional functionality that affects the transitioning between states. If the spin function is called while the Arduino is not in state 1, it will set a global flag to true, indicating that the Arduino has started a movement. If the movement flag is true and the Arduino is back in state 1, this indicates that the machine has just finished a movement. In this case, if the spin function is called it increments the state of the Master Mind, increments the substate of the system, and resets the relevant flags to prepare for the system to send a new movement to the Arduino. This function takes in the current substate and the desired Arduino movement as inputs.

The readPX4 function interprets the global variable PXmsg. It takes no arguments but simply changes the states of the system based on the messages received from the drone. If the message from the PixHawk begins with a warning or error, it will display the appropriate message and reset the program. Otherwise, if the message is relevant to the progress of the calibration, it will change the Master Mind state and set the appropriate flags. In these cases, there are several options for what the message could be. The string "progress <100>" indicates that the current calibration has been completed. In this case the done flag will be set to true and the state will be decremented by one if Master Mind is in the "hold" stage. The "side done" message acts largely the same as the "progress <100>" message but sets the sideDone flag rather than the done flag. This message corresponds to the magnetometer and accelerometer calibrations, as they must be calibrated on multiple sides before being considered fully calibrated. If the message contains the string "7 s", the PixHawk is prompting the user to spin the drone for seven seconds. This message only appears when calibrating the magnetometer. The flag spinReady is set, indicating that the primary axis is actively spinning, and Master Mind waits in the spin state before moving back to the orient state to reorient.

5.First Time Setup

5.1.Hardware

1. Set the structure on a level surface. Raise or lower the base's adjustable feet and check the levelness of the base's top surface with a bubble level.
2. Fence out a taped area on the ground around the system indicating where the operator should not enter to avoid colliding with the system.
3. Outside of this taped area, set up a desk and place the Linux computer, monitor, and emergency stop button on it.
4. Place the drone into the drone plate and secure it using the straps.
5. Plug in the power supply. (One side into a power outlet, the other into the female barrel connector near the Arduino.)
6. Plug in the Arduino USB cord into the Arduino (inside the base) and into the Linux computer.
7. Place aircraft into drone mounting plate, as seen in Figure 8.

Figure 8: Inserting the Drone

8. Secure the drone to the plate using velcro straps, as seen in Figure 9.

Figure 9: Strapping Down the Drone

5.2.Software

The required software can be found in the "Drone Calibration Software" folder. The software may need the following to run properly.

System Requirements:
Computer:
Python 3.8+ (Tkinter)
Linux
Python Libraries: TKinter, Pymavlink, serial

Adjustable Parameters:

Arduino:

sensorOffset[#] - This is the number of steps that each motor must take from the Reed switch to the start spot, which should be completely level. The number corresponds to which axis, i.e. sensorOffset1 corresponds to the primary axis. When moving the calibrator to a new location, these should be calibrated. We recommend using the gyroscope or level horizon calibration to run through the locating Reed switches state, then using the bubble level to find the correct number of steps to end up level.

step7 - This is the numbers of steps the primary motor takes which should be around 7 seconds long. This value can be changed to lengthen or shorten the amount of time the primary spins for. It should be around 7000 steps.(NOTE: Changing Dlay will affect how many steps are required for 7 seconds).

Dlay - For developers, this variable controls the delay between steps of the stepper motors. The primary motor should be around 1000 microseconds, and the secondary and tertiary should be 500 microseconds. (NOTE: changing this value with change the speed at which the motor runs)

rampTime - For developers, rampTime is the number of **steps** involved in the linear ramp that increases the speed of the motor from stopped. It should be around 250 steps.

Python:

ardAddress - Arduino address is on line 16 of MasterMind.py. This is the port of the arduino and it is usually something like /dev/ttyACM0, but may need to be adjusted.

PXaddress - PixHawk address is on line 17 of MasterMind.py. This is the port of the PixHawk telemetry modules and it is usually something like /dev/ttyUSB0, but may need to be adjusted.

skip - For developers, skip is on line 15 of MasterMind.py. This variable allows you to turn off the locating sequence at the beginning of all calibrations. Setting it ‘True’ will make it skip this sequence.

Table 1. Global Variables of the Python Code

Global Variable	Purpose
ard2	Bool. Indicates whether the Arduino has entered state 2 or 5. Confirms that the spin command was received and executed by the Arduino. Set once the Arduino exits state 1, and cleared when Arduino enters state 1.
comReady	Bool. Indicates that the Arduino is in state 1 and is ready to receive a spin command. Ensures only 1 command is sent to Arduino. Set when arduinoState is 1, and cleared after a command is sent to the Arduino.
done	Bool. Indicates if a sensor calibration is complete. Allows state to increment, sending Master Mind to the next sensor calibration. Set when calibration is done, and cleared when state is sent to next calibration.
indCalib	Int. Indicates which button has been pressed. 0 for full calibration, 4 for gyroscope, 6 for level horizon, 8 for compass, and 10 for accelerometer (based on the orient states of each sensor).
PXmsg	Str. Contains the last message read from the PixHawk. Cleared before moving to next sensor calibration in the full calibration.
sideDone	Bool. Indicates that a side is done calibrating. Set when PXmsg says the side is calibrated, and cleared when the Master Mind state is decremented back to the orient state for the sensor and once the whole sensor calibration is complete.
spinReady	Indicates if the primary axis is currently spinning during the compass calibration. Set when PXmsg tells us to spin for 7 s, and cleared once the side is done calibrating and the spin is complete (Arduino is in state 1 again).
state	Int. Indicates the state of the Master Mind task. Refer to state diagram in Figure 7. Increments in spin function, decrements in readPX4 function, and is set in buttons and in MasterMind states.
stateX	Int. Indicates the substate of the current Master Mind state. Incremented in spin state and in the Master Mind state.

6.Running a Calibration

1. Power on the Linux computer.
2. Plug in the power supply.
3. Open the “Drone Calibration Software” folder. Open "Drone Calibrator" script.
4. Load the drone into the attachment plate cutout and strap it down securely. (Note, if this step is completed with the app open, the motors will hold the position. This step may be done before step three in order to have movable axes.)
5. Power on the drone.
6. On the app that pops up, there are individual calibration buttons, in green, and the full calibration button, in purple. Select your calibration option and the calibration process will begin. (No further intervention is needed.)
7. Select any other calibrations you may to complete.
8. Once all calibrating of that drone is done, unload it from the calibrator (either selecting quit to close the app to power off the motors or leaving the app and motors on).
9. Close the app by pressing quit.

7.Troubleshooting

Table 2 shows commonly occurring issues and their solutions.

Table 2. Troubleshooting

Possible Error	Solution
Could not open port dev\ttyACM# or rapid output of nonsensical Arduino states in the terminal.	The serial address for the Arduino may be wrong. Check ardAddress in MasterMind.py.
Machine does not run.	Ensure that the emergency stop is in the up position.
Stuck in “connecting to drone”.	Unpower and then power the PixHawk or reset it.
If it stays in welcome state ("Welcome to the amazing new drone calibrator! Loading..." and no readout in the terminal behind the app)	Unplug the Arduino and plug it back in.
If the drone is in the middle of a calibration when the app is turned on, it will either not get out of the "connecting to drone" state or it may begin moving in ways inappropriate for the current calibration.	Restart the PixHawk.

The Arduino has a ~70 min long counter. This may cause issues, but they are not yet defined.	Restart the Arduino.
Developers only: While uploading new Arduino code. Programmer is out of sync.	Unplug the Arduino and plug it back in (beware: address may change).

Design Failure Mode and Effects Analysis

Product: ______________________________ Prepared by: ______________________________

Team: ______________________________

Appendix L: Failure Modes and Effects Analysis (FMEA)

Date: ________________ (orig)

												Action Results			
System / Function	**Potential Failure Mode**	**Potential Effects of the Failure Mode**	**Severity**	**Potential Causes of the Failure Mode**	**Current Preventative Activities**	**Occurence**	**Current Detection Activities**	**Detection**	**Priority**	**Recommended Action(s)**	**Responsibility & Target Completion Date**	**Actions Taken**	**Severity**	**Occurence**	**Criticality**
Structure / Support drone weight	Structure breaks	a) Drone is damaged b) sharp edges exposed c) motors are damaged d) calibration fails	7	1) Material is too weak 2) Fasteners shear 3) Bearing attachments fail 4) Device is unstable	1) Conduct load analysis 2) chose material with strength above loads 3) chose sufficient safety factor 4) weld or fasten joints to handle loads	1	Visual inspection for cracks broken pieces, etc	3	21	Do stress analysis on load bearing components	Tyler, Zach, and Matthew				
	Structure Flexes too much	a) Innacurate calibration b) user not confident in device c) additional stress on motors (wear out)	4	1) Material not stiff 2) Fasteners are loose 3) motors spin too fast	1) choose material with high stiffness 2) ensure rotation is slow enough to not deflect structure during rotation 3) routinely check all fasters	3	look for deflections while calibrating	3	36	Do research on materials, do hand calculations and initial FEA to determine stresses on structure	Matthew, Zach, and Tyler				
	Structure can't rotate while supporting drone	a) calibration not completed	6	1) Too much friction in turntable 2) deflection causes lack of clearance	1) incorporate bearing into rotating table 2) incorporate sufficient clearences into design such that it can make full rotations	1	see if calibration is successful or if device is halted	1	6	Do stress and clearance analysis and test structure	Tyler and Matthew				
									0						
Attachment / Secure drone	Drone falls out of device	a) Drone is damaged b) User has to reload drone c) rest of device is damaged	6	1) clamp/strap not secured correctly by user 2) attachment not secure in all orientations 3) Attachment material weak or brittle	1) ensure drone is fully constrained in all directions 2) attachment material is strong enough to support drone 3) provide instructions for how to correctly load drone	2	visual inspection	1	12	Analyze attachment mechanism for stress and constraining all degrees of freedom	Tyler and Matthew				
	Drone can't be easily loaded into attachment	a) User is frustraded b) Time is wasted	2	1) Non ergonomic design 2) attachment is difficult to access inside device	1) make attachment easily accesible 2) provide instructions for easy loading techniques	5	ergonomic testing	2	20	Do preliminary testing with attachment method prototypes	Tyler, Jackie, and Matthew				
	Attachment interferes with sensor calibration	a) Calibration is unsucessful	6	1) bad material is chosen	1) don't use materials that interfere with sensors	3	flight log data for sensor interference	2	36	Continue material testing for interference	Tyler				
Attachment / Reduce movement (vibration)	Drone can move within the attachment	a) Drone is damaged b) calibration fails c) calibration is inaccurate	4	1) friction between surfaces too low 2) clamping force is too low 3) attachment not stiff 4) drone loaded incorrectly	1) add non slip surface material 2) ensure device can attain adaquate holding force 3) provide correct loading instructions	7	Flight log data for vibration	2	56	Prototype attachment methods for drone and test these using the flight log data	Matthew and Tyler				
									0						
Electrical / drive motors	Motors run at too high of speed	a) User may be injured b) User doesn't trust device stability c) Calibration not completed with fast movement	9	1) motors not geared correctly 2) controller is unstable 3) motors not compatable with device	1) choose correct type of motors 2) design appropriate gear ratio if necessary	2	watch device go through calibration process	1	18	Test motors with equivalent loads	Zach				
	Controller does not accurately locate motor positions	a) Calibration is inaccurate b) calibration fails c) Device must be reset	6	1) angle measurement is not correct	1) choose motors that can accurately locate their own rotational position	1	external measurement to compare to devices perceived location	5	30	Utilize multiple angle measuring devices	Jackie, Ryan and Zach				
Electrical / Transmit signals	Wiring does not power motors	a) Device doesn't turn on b) Motors do not spin c) calibration fails	5	1) Device not pluged in 2) motors not connected 3) faulty wires	1) secure all electrical connections	3	see if device has any power	2	30	Buy comercial parts and standard connectors	Jackie, Ryan and Zach				
									0						

Design Failure Mode and Effects Analysis

Product: ______________________________ Prepared by: ______________________________

Team: ______________________________ Date: ________________ (orig)

												Action Results			
System / Function	**Potential Failure Mode**	**Potential Effects of the Failure Mode**	**Severity**	**Potential Causes of the Failure Mode**	**Current Preventative Activities**	**Occurence**	**Current Detection Activities**	**Detection**	**Priority**	**Recommended Action(s)**	**Responsibility & Target Completion Date**	**Actions Taken**	**Severity**	**Occurence**	**Criticality**
Electrical / Allow for rotation	Motors can't be driven due to loss of connection	a) Calibration stops after number of roations b) Wires become tangled c) wiring breaks during rotation d) open electrical circuits	6	1) slip rings not installed correctly	1) use slip rings where required	3	rotate motors and check connection	2	36	Purchase slip rings to test with our next prototype. Also develop a "calibration path" so we know how to do wire-routing	Ryan and Zach				
									0						
Software / Receive calibration outputs	Device can't read calibration outputs	a) device does not take any action b) User has to manually change device motion c) Calibration fails	6	1) code written poorly	1) learn about QGroundControl software and interfacing	2	debug	3	36	Research QGroundControl and start prototyping code structure	Jackie and Ryan				
									0						
Software / Interface Controller and device	Software doesn't control motors	a) motors spin irradically b) motors don't spin c) calibration fails	6	1) software outputs incorrect voltages	1) write robust code	1	debug	3	18	Organize and comment code well	Jackie and Ryan				
	Software has bugs	a) user does not know what went wrong b) device behaves unpredictablly	4	1) software not debugged	1) do extensive debugging	5	debug	3	60	Have a software "prototyping" plan and start getting base code written	Jackie, Ryan and Zach				

Appendix M: Indented Bill of Materials

Assembly Level	Part Number		Description			Vendor	Quantity	Cost	Total Cost	Notes
		Lvl0	*Lvl1*	*Lvl2*	*Lvl3*					
0		Final Assy								
1	**100**		**Base Structure**							
2				Primary Joint						
3	506				Nema 34 w/ 20:1	Stepper Online	1	$126.67	$126.67	
3	102				Slip ring 12-wire	CNTBR (through Amazon)	1	$41.35	$41.35	
3	103				Turntable	McMaster	1	$350.40	$350.40	
3	104				Shaft Support	McMaster	1	$55.34	$55.34	
3	105				Keyed Flanged Hub	MRO	1	$11.50	$11.50	
3	106				Primary Shaft	McMaster	1	$28.16	$28.16	
3	107				Key	McMaster	1	$10.14	$10.14	
3	108				Adapter Bushing 12-14mm	McMaster	1	$7.79	$7.79	
3	109				15mmx15mm Coupling	McMaster	1	$65.39	$65.39	
3	909				M5 x.8 x25mm Coarse Thread Bolts (50 Pk)	McMaster	6	*	N/A	
3	910				M5 x .8 Locknut Coarse (100 Pk)	McMaster	6	*	N/A	
3	911				M5x.8x25 Flat Head Screws (50 Pack)	McMaster	4	*	N/A	
3	902-5				Turn Table Mounting Plate Base	Amazon	1	*	N/A	Custom Machined
3	902-6				Turn Table Mounting Plate Fork	Amazon	1	*	N/A	Custom Machined
3	902-7				Flanged Hub Adaptor	Amazon	2	*	N/A	Custom Machined
3	914-1				Base Proximity Sensor Mount V1	Amazon	1	*	N/A	3D Printed
3	914-4				Driver Mounting Plate	Amazon	1	*	N/A	3D Printed
3	914-5				Primary Motor Mounting Plate	Amazon	1	*	N/A	3D Printed
2				80/20 Frame						
3	8020-1				Turntable Horizontal Supports (L=24")	8020.net	6	*	N/A	
3	8020-2				Turntable Horizontals long (L=22")	8020.net	4	*	N/A	
3	8020-3				Turntable Horizontals short (L=20")	8020.net	4	*	N/A	
3	8020-4				Turntable Verticals Supports (L=9")	8020.net	4	*	N/A	
3	8020-5				Corner Verticals Supports (L=8")	8020.net	4	*	N/A	
3	801				Frame Corner Fastener (3x3)	McMaster	16	*	N/A	
3	802				Gusset Brackets 1"	McMaster	24	*	N/A	
3	110				Leveling Foot	McMaster	4	$6.19	$24.76	
1	**200**		**Rotating Fork**							
2				Secondary Joint (motor)						
	505				Nema 17 w/ 100:1	Stepper Online	2	$55.61	$111.22	
3	904				Flange Bearing	VXB	2	*	N/A	
3	901				Hex Shaft	WCP	2	*	N/A	
3	*903*				Shaft Coupling	WCP	2	*	N/A	
3	903				Hex Clamping Hub	Amazon				
3	902-1				NEMA 17 Mounting Plate	Amazon	4	*	N/A	Custom Machined
3	902-3				Slip-Ring Bearing Support Plate	Amazon	4	*	N/A	Custom Machined
3	207				Slip Ring 6-wire	Amazon	1	$41.35	$41.35	
3	902-4				Hex Shaft to Ring Adapter	Amazon	2	*	N/A	Custom Machined
3	905				Flat Head Screw, 6-32, 11/16"	McMaster	8	*	N/A	
3	906				Flat Head Screw, M4 x 0.7 mm, 18 mm	McMaster	8	*	N/A	
3	914-2				Fork Proximity Sensor Mount V1	Amazon	1	*	N/A	3D Printed

Assembly Level	Part Number		Description			Vendor	Quantity	Cost	Total Cost	Notes
2				80/20 Frame						
3	8020-6				Fork Arms (L=33.5")	8020.net	4	*	N/A	
3	8020-7				Fork Base (L=50.5")	8020.net	2	*	N/A	
3	8020-10				Fork Corner Suppports (L=12")	8020.net	4	*	N/A	
3	801				Frame Corner Fastener (3x3)	McMaster	4	*	N/A	
3	802				Gusset Brackets 2"	McMaster	4	*	N/A	
3	805				1/4"-20 Bolt	McMaster	24	*	N/A	
3	806				T-nut Nut	McMaster	24	*	N/A	
3	804				Gusset Bracket 3x1"	McMaster	2	*	N/A	
3	807				Diagonal Brace Ends	McMaster	2	$17.60	$35.20	
1	**300**		**Inner Ring**							
2				Tertiary Joint (motor)						
3	505				Nema 17 w/ 100:1	Stepper Online	2	$44.80	$89.60	
3	904				Flange Bearing	VBX	2	*	N/A	
3	901				Hex Shaft	WCP	2	*	N/A	
3	902-1				NEMA 17 Motor Mounting Plate	Amazon	4	*	N/A	Custom Machined
3	902-2				Bearing Mounting Plate	Amazon	4	*	N/A	Custom Machined
3	403				3x1x1.7" Aluminum Billet	Cal Poly IME Department	2	*	N/A	Custom Machined
3	914-3				Ring Proximity Sensor Mount V1	Amazon	1	*	N/A	3D Printed
2				80/20 Frame						
3	8020-9				Ring Arms (L=42.5")	8020.net	4	*	N/A	
3	8020-8				Ring Motor Supports (L=3")	8020.net	4	*	N/A	
3	803				Gusset Brackets 2"	McMaster	4	*	N/A	
3	801				Frame Corner Fastener	McMaster	8	*	N/A	
3	905				Flat Head Screw, 6-32, 11/16"	McMaster	24	*	N/A	
3	906				Flat Head Screw, M4 x 0.7 mm, 18 mm	McMaster	24	*	N/A	
1	**400**		**Drone Attachment**							
2				Drone Attachment Plate						
3	401				Straps	McMaster	1	$2.63	$2.63	
3	901				Hex Shaft	WCP	1	*	N/A	Already in Ring Assembly
3	402				Custom Plate	Amazon	1	*	N/A	Custom Machined
3	403				Plate Mounting Bracket	Cal Poly IME Scrap	2	$0.00	$0.00	Custom Machined
3	404				Bubble Level	McMaster	1	$3.22	$3.22	
3	912				1/4" Washer	McMaster	8	*	N/A	
3	905				6/32 x 2" Socket Head Screws	McMaster	8	*	N/A	
3	805				1/4"-20 Bolt	McMaster	8	*	N/A	
3	913				1/4" Locknut	McMaster	8	*	N/A	
3	920				Drone Padding	Amazon				
2				Attachment Shaft Coupling						
3	402				T Clamps	WCP	4	$19.99	$79.96	
3	903				Hex Clamping Hub	Amazon				
3	*903*				Shaft Coupling	WCP	1	*	N/A	
1	**500**		**Electronic Hardware**							

Assembly Level	Part Number		Description			Vendor	Quantity	Cost	Total Cost	Notes
2	501			Motor Driver 1.0-4.2A		Stepper Online	2	$26.66	$53.32	
2	502			Motor Driver 2.4-7.2A		Stepper Online	1	$23.64	$23.64	
2	503			Arduino		Amazon	1	$18.00	$18.00	
2	504			PowerSupply		Amazon	1	$39.99	$39.99	
2	507			USB Cable		Amazon	1	$1.11	$1.11	
2	508			Proximity Sensor Magnet		Newark	3	$3.15	$9.45	
2	509			Proximity Sensor		Newark	3	$3.14	$9.42	
***Multi-assembly parts shown below**										
	800		**8020 Parts**							
	801				Frame Corner Fastener	McMaster	12	$7.62	$91.44	
	802				Gusset Brackets 1"	McMaster	8	$6.54	$52.32	
	803				Gusset Brackets 2"	McMaster	8	$9.99	$79.92	
	804				Gusset Bracket 3x1"	McMaster	2	$9.46	$18.92	
	805				1/4"-20 Bolt	McMaster	150	$0.14	$21.24	
	806				T-nut Nut	McMaster	100	$0.21	$21.00	
	900		**Miscellaneous**							
	901				Hex Shaft Stock	WCP	2	$24.21	$48.42	
	902				Sheet Aluminum 1/4" Thick, 12"x12"	Amazon	4	$9.41	$37.64	Custom Machined
	903				Shaft Coupling	WCP	5	$15.98	$79.90	
	904				Flange Hex Bearing	WCP	4	$8.98	$35.92	
	905				Flat Head Screw, 6-32, 11/16"	McMaster	100	$0.09	$8.60	
	906				Flat Head Screw, M4 x 0.7 mm, 18 mm	McMaster	100	$0.14	$13.62	
	907				Round Bearing	WCP	2	$4.49	$8.98	
	908				Hex Hub Clamp	WCP	4	$7.99	$31.96	
	909				M5 x.8 x25mm Coarse Thread Bolts (50 Pk)	McMaster	1	$7.74	$7.74	
	910				M5 x .8 Locknut Coarse (100 Pk)	McMaster	1	$6.46	$6.46	
	911				M5x.8x25 Flat Head Screws (50 Pack)	McMaster	1	$9.66	$9.66	
	912				1/4" Washer	McMaster	1	$9.60	$9.60	
	913				1/4" Locknut (25 pk)	McMaster	1	$3.22	$3.22	
	914				PolyMax Polycarbonate Filament	Amazon	1	$39.99	$39.99	
	915				Emergency Stop	Amazon	1	$10.29	$10.29	
	916				3/4" 6-32 Low profile screws	McMaster	1	$10.73	$10.73	
	917				M8x1.25 30mm Flat Head Screws	McMaster	1	$5.86	$5.86	
	918				1" 1/4"-20 Srews	McMaster	1	$2.35	$2.35	
	919				1/2" #4 UNC Wood Screws	McMaster	1	$4.53	$4.53	
	8020-#				T-Slot Rail, Silver, 1" High x 1" Wide, 10ft	McMaster	6	$30.54	$183.24	Cut to length
						Total Parts	862	**Total Cost**	$2,083.16	

NOTE: for accurate total cost of purchased parts, see Master sheet
These costs do not reflect shipping, tax, etc

Appendix N: Budget

CYD Item No.	Item Number	Description	Quantity	Price	Amount	Source	Link	Status	Notes	Shipping + Tax	Date Ordered	Total Amount	$3,290.44
505	17HS15-1684S-H(	100:1 GB, Nema 17, 5500 oz-in	2	$60.01	$120.02	StepperOnline	https://www	Arrived	Secondary and Tertiary Motors	$ 17.85	1/26/2020	First Order Total	$1,054.19
502	DM860Y	Digital Stepper Driver 2.4-7.2A AC18V-80V / DC 36V-110V	2	$42.36	$84.72	StepperOnline	https://www	Arrived	Big Motor Drivers				
902	OL-400730-12X12	Sheet Aluminum 1/4" Thick, 12"x12"	1	$9.41	$9.41	Amazon	https://www	Arrived	For milling custom plates	$ 5.37	1/26/2020		
207	SRT012	Slip Ring - 6 Wire, 10A	1	$41.35	$41.35	Amazon	https://www	Arrived	1/2" ID, may need to sand ID/OD of Hex shaft to make fit (1mm interference)				
903	545674	1/2 inch Bore Hex Clamping Hub	2	$15.98	$31.96	Amazon	https://www	Arrived	Connection between hex shaft and inner ring	$ 6.99	1/26/2020		
904	217-3875	1/2" Hex Flanged Bearing	2	$8.98	$17.96	WCP	https://www	Arrived	Press fit into custom plates	$ 17.66	1/26/2020		
901	217-3309	1/2" Hex Shaft (18")	2	$17.98	$35.96	WCP	https://www	Arrived	1/2" flat to flat (13.75mm OD with rounded corners)				
907	217-4006	1/2" ID Round Flanged Bearing	2	$4.49	$8.98	WCP	https://www	Arrived	Press fit into custom plates	$ 10.22	1/26/2020		
806	3382	1/4-20 Slide-in T-Nut	100	$0.21	$21.00	8020.net	https://8020	Arrived	Were not in stock on McMaster/Amazon	$ 43.91	1/26/2020		
8020	47065T101	T-Slot Rail, Silver, 1" High x 1" Wide, 10ft	6	$30.54	$183.24	McMaster	https://www	Arrived	10 ft lengths	$ 86.04	1/27/2020		
801	47065T267	Corner Surface 2" Bracket for 1" 8020	12	$7.62	$91.44	McMaster	https://www	Arrived	For fork and inner ring (larger 3" not in stock)	$ 28.49	1/27/2020		
805	91255A540	Button Head Screw 1/4"-20 Thread, 3/4"	50	$0.19	$9.46	McMaster	https://www	Arrived	For mounting larger thicknesses to 8020				
802	47065T663	Gusset Bracket, 1" Long for 1" 8020	8	$6.54	$52.32	McMaster	https://www	Arrived	90 degree internal corners, 1"				
803	47065T736	Gusset Bracket, 2" Long for 1" 8020	4	$9.99	$39.96	McMaster	https://www	Arrived	90 degree internal corners, 2" (For inner ring)				
805	91355A178	Flanged Head Screw, 1/4"-20 Thread, 5/8"	20	$0.76	$15.16	McMaster	https://www	Arrived	For mounting custom plates to 8020				
905	92210A164	Flat Head Screw, 6-32, 11/16"	100	$0.09	$8.60	McMaster	https://www	Arrived	For mounting Hex Hubs to inner ring				
N/A	4973A51	Ultra-Grip T-Handle Hex Key, 5/64" Drive	2	$2.59	$5.18	McMaster	https://www	Arrived	For screwing in hex hub screws				
906	92125A196	Flat Head Screw, M4 x 0.7 mm, 18 mm	100	$0.14	$13.62	McMaster	https://www	Arrived	For mounting motors to custom plates				
805	91255A537	Button Head Screw, 1/4"-20 Thread, 1/2"	150	$0.14	$21.24	McMaster	https://www	Arrived	Standard 8020 screw				
804	3136N175	Corner Bracket for 3" 8020, 1"	2	$9.46	$18.92	McMaster	https://www	Arrived	For internal corners on fork	$ 7.16	1/27/2020		
401	3955t291	Velcro Straps, 48" long 1" wide	2	$3.61	$7.22	McMaster	https://www	Arrived		$ 151.02	2/23/2020	Second Order Total	$2,007.41
805	92196A305	1/4-20 x 1 7/8" Socket Head Screws (25 Pk)	1	$12.62	$12.62	McMaster	https://www	Arrived	Holding Blocks of Aluminum to polycarbonate plate				
805	92210A539	1/4-20 x 5/8" Flat Head Screws (50 pk)	1	$6.68	$6.68	McMaster	https://www	Arrived	Holds custom turn table plates to 8020				
402	8574k66	Polycarbonate Sheet 24"x 48" x .5"	1	$174.65	$174.65	McMaster	https://www	Arrived					
8020	47065t101-47065	T-Slot Rail, Silver, 1" High x 1" Wide, 8ft	2	$23.57	$47.14	McMaster	https://www	Arrived					
103	8700K200	Primary Turntable, 670 lbs	1	$379.56	$379.56	McMaster	https://www	Arrived					
110	47065t341	Anti-Slip Leveling Foot	4	$6.19	$24.76	McMaster	https://www	Arrived					
106	1439k26	Keyed Shaft 15mm x 300mm (Primary Shaft)	1	$28.16	$28.16	McMaster	https://www	Arrived	Turn one side down for Slip Ring, use bushing to get diameter back to 14mm				
107	96717a160	5x5x16mm Machine Key - 10 Pack	1	$10.14	$10.14	McMaster	https://www	Arrived					
905	92196A159	6/32 x 2" Socket Head Screws (25 Pack)	1	$4.33	$4.33	McMaster	https://www	Arrived	For mounting hex hub to plate blocks				
912	92917A155	1/4" Washers (100 Pack)	1	$9.60	$9.60	McMaster	https://www	Arrived	For polycarbonate plate, etc				
909	91292A129	M5 x.8 x25mm Coarse Thread Bolts (50 Pk)	1	$7.74	$7.74	McMaster	https://www	Arrived	For outer turntable				
910	93625A200	M5 x .8 Locknut Coarse (100 Pk)	1	$6.46	$6.46	McMaster	https://www	Arrived	For outer turntable				
911	92125A216	M5x.8x25 Flat Head Screws (50 Pack)	1	$9.66	$9.66	McMaster	https://www	Arrived	For Inner turntable				
913	90630A110	1/4" Locknut (25 pk)	1	$3.22	$3.22	McMaster	https://www	Arrived	For polycarbonate plate, etc				
801	47065T267	3x3 Corner Surface Bracket	16	$7.62	$121.92	McMaster	https://www	Arrived					
802	47065T663	1" Corner Bracket	28	$6.54	$183.12	McMaster	https://www	Arrived					
109	2469K75	15mmx15mm Coupling	1	$65.39	$65.39	McMaster	https://www	Arrived					
501	DM542Y	Digital Stepper Driver 1.0-4.2A DC20V-50V	2	$26.66	$53.32	StepperOnline	https://www	Arrived	Small Drivers	$ 11.06	2/20/2020		
506	34HS38-4004D-S(	20:1 Spur GB, Nema 34, 12,700 oz-in	1	$126.67	$126.67	StepperOnline	https://www	Arrived	Primary Motor	$ 60.15	2/20/2020		
904	217-3875	1/2" Hex Bearing	4	$4.49	$17.96	WCP	https://www	Arrived		$ 10.87	2/18/2020		
105	H14MM	14mm Keyed Flange Hub	1	$11.50	$11.50	MRO	https://www	Arrived		$ 9.65	2/20/2020		
504	BX-24010000	Power Supply, 24V 10A	1	$39.99	$39.99	Amazon	https://www	Arrived	Cord Included, 24V 10A	$ 19.61	2/18/2020		
502	A000066	Arduino Uno	1	$18.00	$18.00	Amazon	https://www	Arrived	Not Ryan's				
102	B07XHQLJ2H	Slip Ring - 12 Wire 10 A	1	$72.25	$72.25	Amazon	https://www	Arrived					
108	a18072700ux005	12mm to 14mm Adapter Bushings	1	$7.79	$7.79	Amazon	https://www	Arrived					
801	47065T267	Corner Surface 2" Bracket for 1" 8020	16	$7.62	$121.92	McMaster	https://www	Arrived	Additional corner brackets for base structure	$ 16.66	2/24/2020		
902	OL-400730-12X12	1/4" 12"x 12" 6061 Aluminum	4	$27.95	$111.80	Amazon	https://www	Arrived					
908	545674	1/2" Hex Hub Clamp	2	$7.99	$15.98	Amazon	https://www	Arrived		$ 8.77	2/18/2020		
507	EMPR01	USB 2.0 A-B Cord, 10 ft	1	$1.11	$1.11	Amazon	https://www	Arrived	10 ft, 2.0	$ 0.09	2/18/2020		
508	AS201901	Proximity Sensor Magnet	3	$3.15	$9.45	Newark	https://www	Arrived					
509	MP201901	Proximity Sensor	3	$3.14	$9.42	Newark	https://www	Arrived					
403	N/A	Billet's of Aluminum	2	$0.00	$0.00	Cal Poly IME Scrap		Arrived				Final Order Total	$228.84
915	43217-5544	Emergency Stop	1	$10.29	$10.29	Amazon	https://www	Arrived		$ 0.80	4/27/2020		

CYD Item No.	Item Number	Description	Quantity	Price	Amount	Source	Link	Status	Notes	Shipping + Tax	Date Ordered	Total Amount	$3,290.44
920	N/A	Drone Plate Padding	1	$12.77	$12.77	Amazon	https://www	Arrived		$ 0.99	4/30/2020		
914	PM70494	PolyMax Polycarbonate Filament	1	$39.99	$39.99	Amazon	https://www	Arrived		$ 2.99	5/15/2020		
918	91251A542	1" 1/4"-20 Srews	1	$2.35	$2.35	McMaster	https://www	Arrived	The link is effectively what we need, not exactly what we bought	$ 0.64			
919	90031A110	1/2" #4 UNC Wood Screws	1	$4.53	$4.53	McMaster	https://www	Arrived	The link is effectively what we need, not exactly what we bought				
N/A	7122A17	7/64" Allen Key	1	$0.18	$0.18	McMaster	https://www	Arrived		$ 9.80	4/28/2020		
916	92220A145	3/4" 6-32 Low profile screws	1	$10.73	$10.73	McMaster	https://www	Arrived					
917	91294A287	M8x1.25 30mm Flat Head Screws	1	$5.86	$5.86	McMaster	https://www	Arrived					
404	2198A85	Bubble level	1	$13.90	$13.90	McMaster	https://www	Arrived					
N/A	5374A15	5/32" T-handle Hex Key	1	$5.13	$5.13	McMaster	https://www	Arrived					
804	47065T511	T-Slotted Framing, Triple Rail	1	$20.87	$20.87	McMaster	https://www	Arrived		$ 16.62	5/11/2020		
807	47065T331	Diagonal Brace Ends	4	$17.60	$70.40	McMaster	https://www	Arrived					

APPENDIX O:

Drawing Packet

Table of Contents

Custom Part Drawings:

McMaster Part Drawings:

47065T736_T-SLOTTED FRAMING

47065T977_ALUMINUM T-SLOTTED FRAMING EXTRUSION

90031A110_ZINC PLATED STEEL FLAT HEAD SCREW

90630A110_HIGH-STRENGTH STEEL NYLON-INSERT LOCKNUT

91251A542_ALLOY STEEL SOCKET HEAD SCREW

91255A537_BUTTON HEAD HEX DRIVE SCREW

91255A540_BUTTON HEAD HEX DRIVE SCREW

91292A129_TYPE 18-8 SS SOCKET HEAD CAP SCREW

91355A178_FLANGED BUTTON-HEAD SOCKET CAP SCREW

92125A196_TYPE 18-8 SS FLAT-HEAD SOCKET CAP SCREW

92125A216_TYPE 18-8 SS FLAT-HEAD SOCKET CAP SCREW

92196A159_18-8 STAINLESS STEEL SOCKET HEAD CAP SCREW

92196A305_18-8 STAINLESS STEEL SOCKET HEAD CAP SCREW

92210A164_18-8 SS HEX DRIVE FLAT HEAD SCREW

92210A539_18-8 STAINLESS STEEL FLAT-HEAD SCKT CAP SCREW

92917A155_NICKEL-PLATED BRASS FLAT WASHER

93625A200_18-8 STAINLESS STEEL NYLON-INSERT LOCKNUT

96717A160_METRIC MACHINE KEY

Purchased Component Specification Sheets:

Arduino UNO

NEMA 17-100GB

NEMA 34-20GB

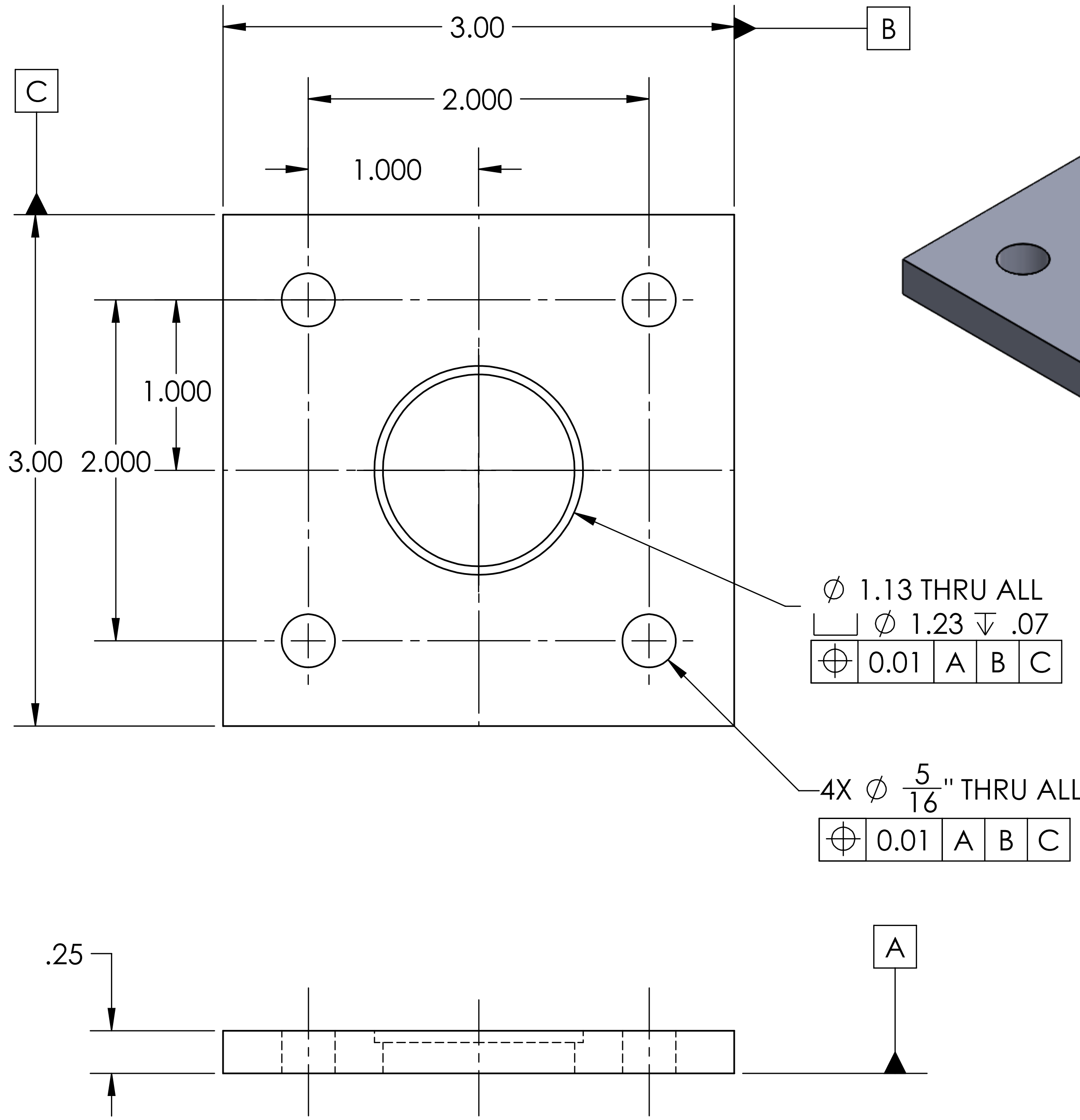
Power Supply Adapter

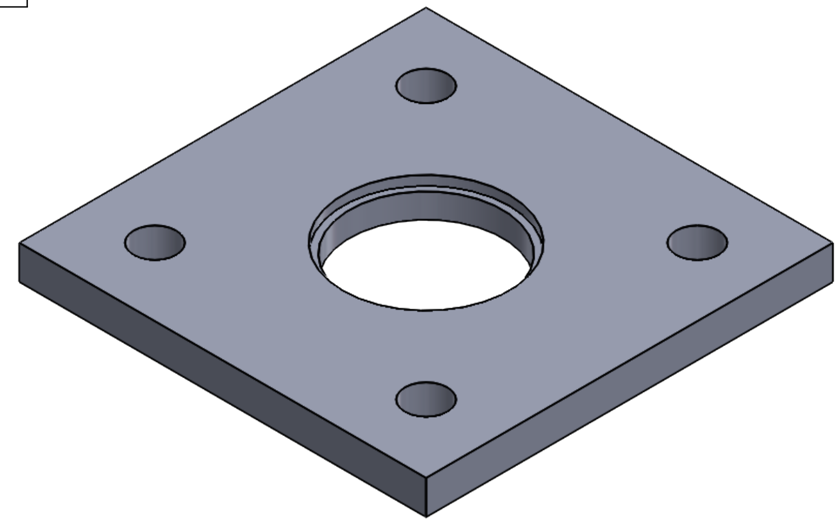
Reed Sensor

Slip Ring 6-wire

Slip Ring 12-Wire

Stepper Drivers

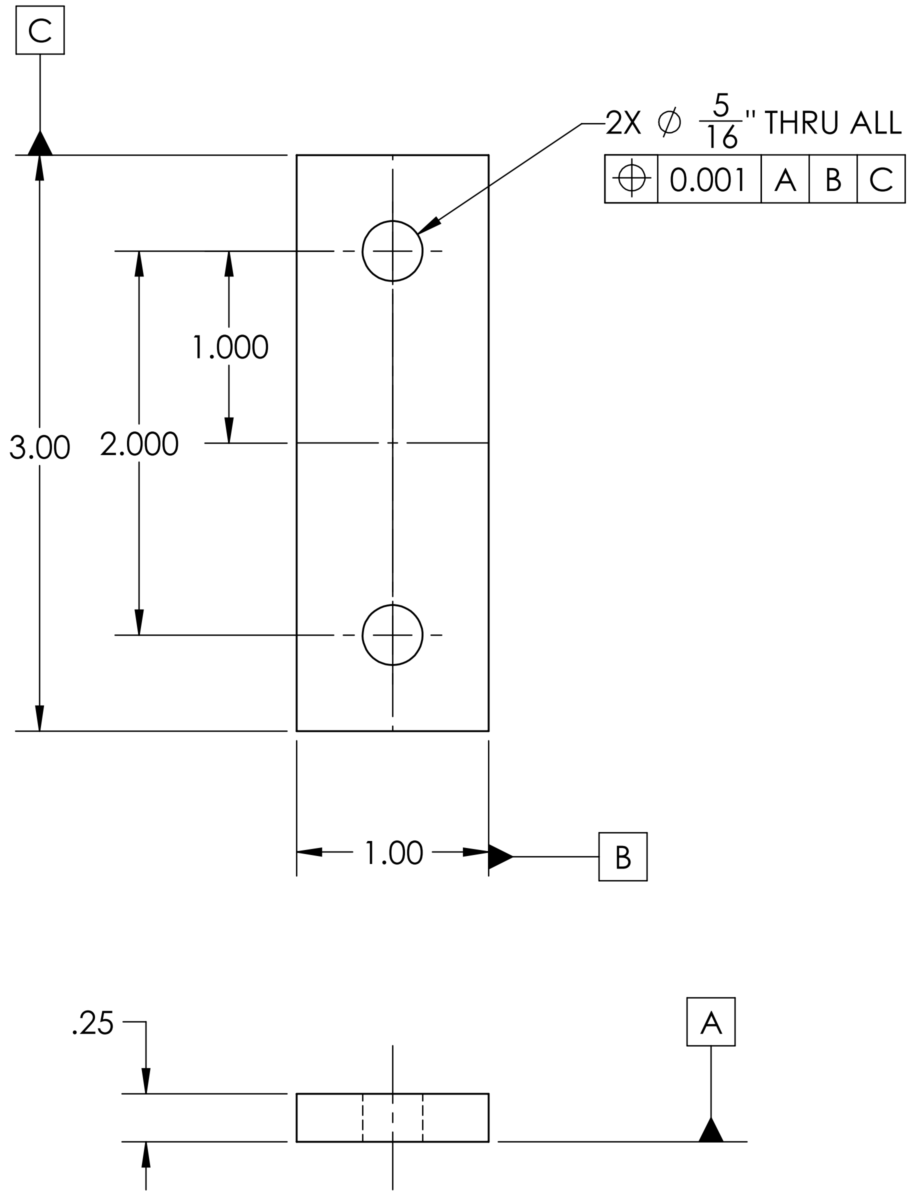




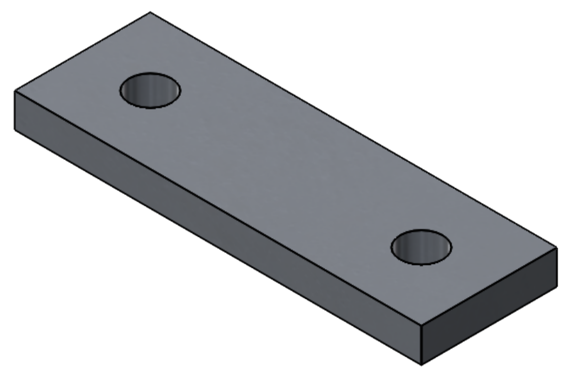
UNLESS OTHERWISE SPECIFIED

1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01

Cal Poly Mechanical Engineering	Lab Section:	Title: Bearing Mounting Plate	Drwn. By: Matthew Carlson
ME 430 - SPRING 2020	Dwg. #: 902-2	Date: 06/03/2020 — Scale: 1:1	Chkd. By: Jackie Paik

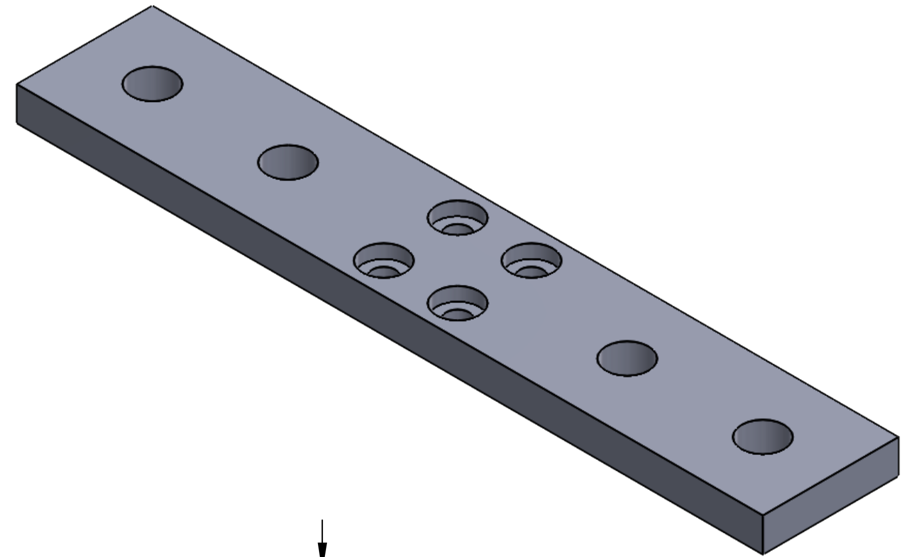


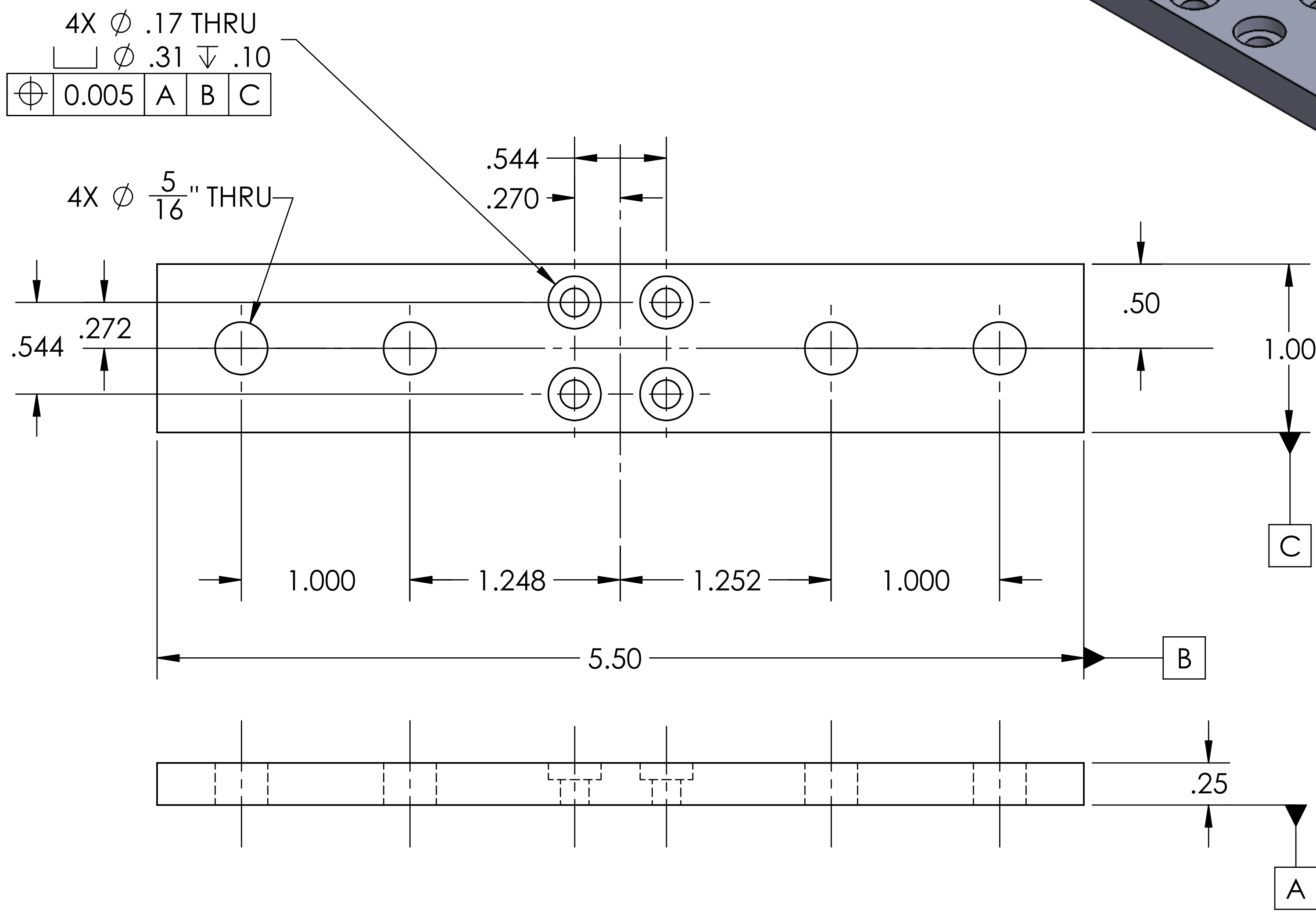
UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01



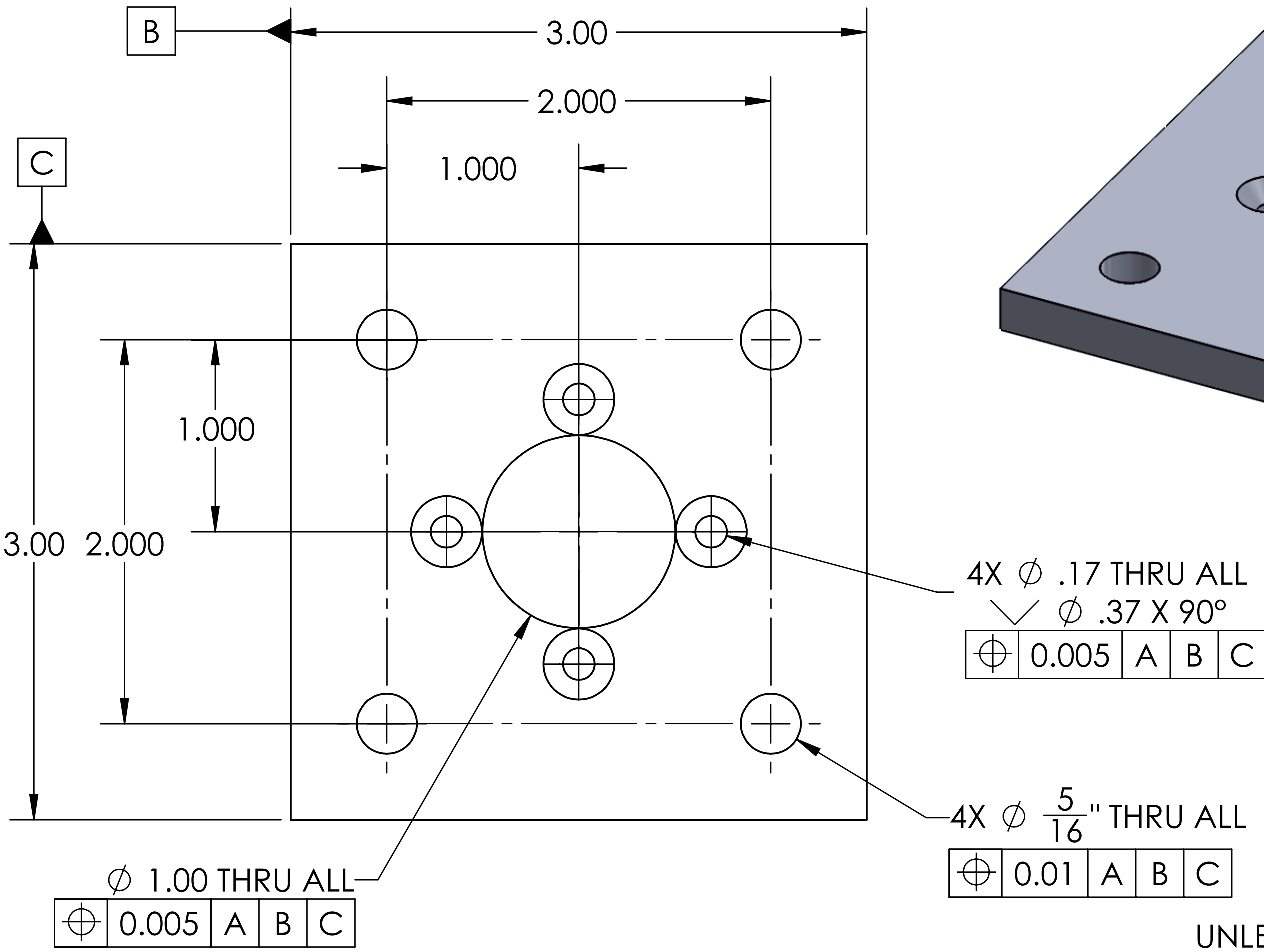
Cal Poly Mechanical Engineering	Lab Section:	Title: Flanged Hub Adapter		Drwn. By: Matthew Carlson
ME 430 - SPRING 2020	Dwg. #: 902-7	Date: 06/03/2020	Scale: 1:1	Chkd. By: Jackie Paik

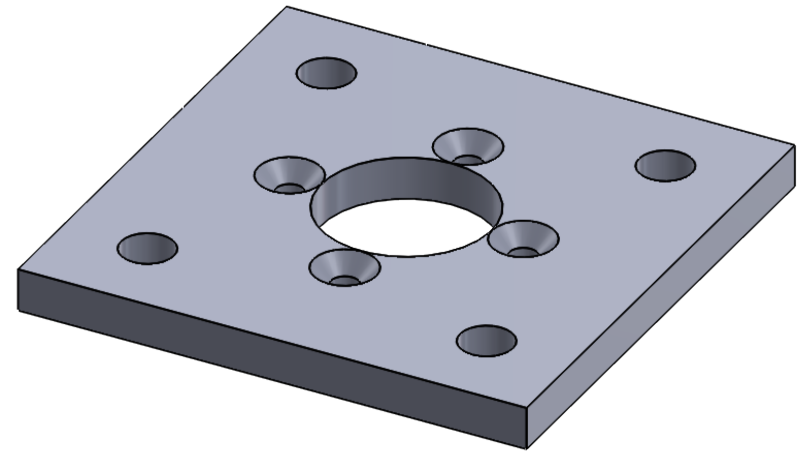
UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01

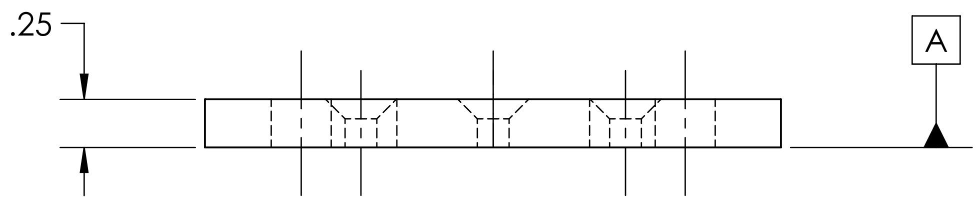




Cal Poly Mechanical Engineering ME 430 -SPRING 2020	Lab Section:	Title: Inner Ring Shaft Mounting Plate		Drwn. By: Matthew Carlson
	Dwg. #: 902-4	Date: 06/03/2020	Scale: 1:1	Chkd. By: Jackie Paik

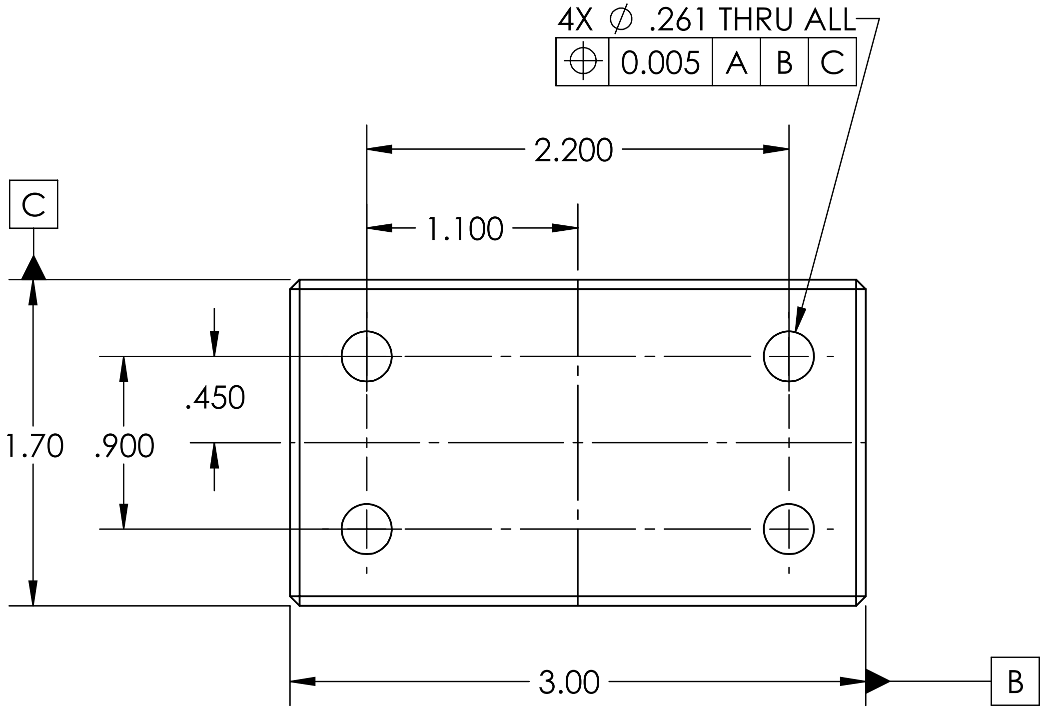


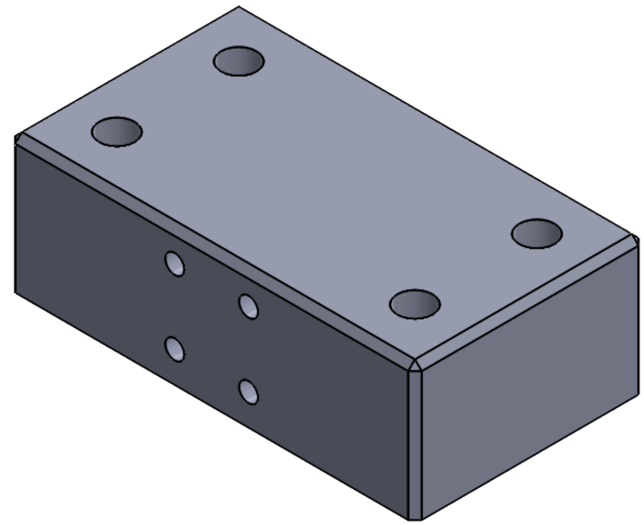


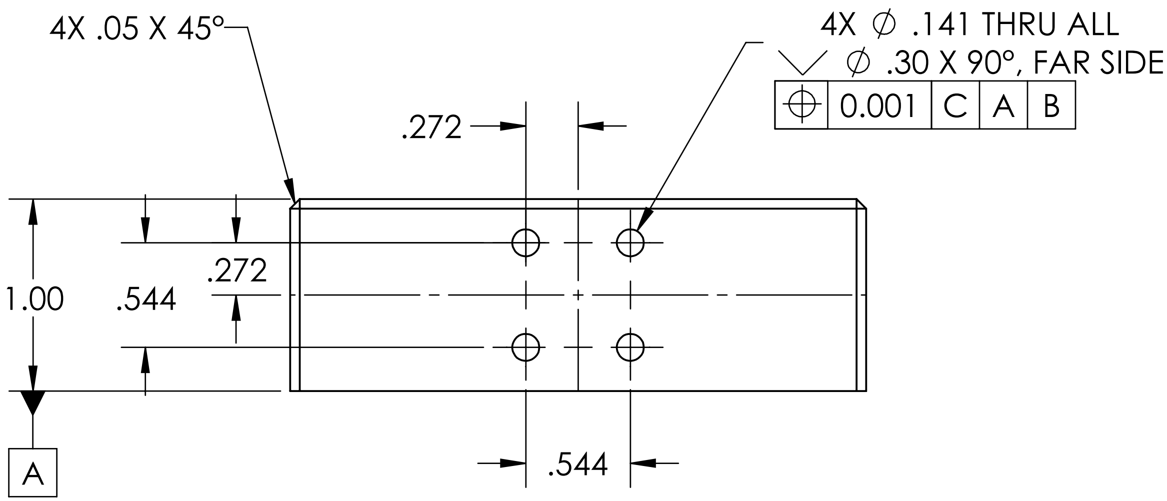


UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01

Cal Poly Mechanical Engineering	Lab Section:	Title: Nema 17 Motor Mounting Plate	Drwn. By: Matthew Carlson
ME 430 - SPRING 2020	Dwg. #: 902-1	Date: 06/03/2020 Scale: 1:1	Chkd. By: Jackie Paik

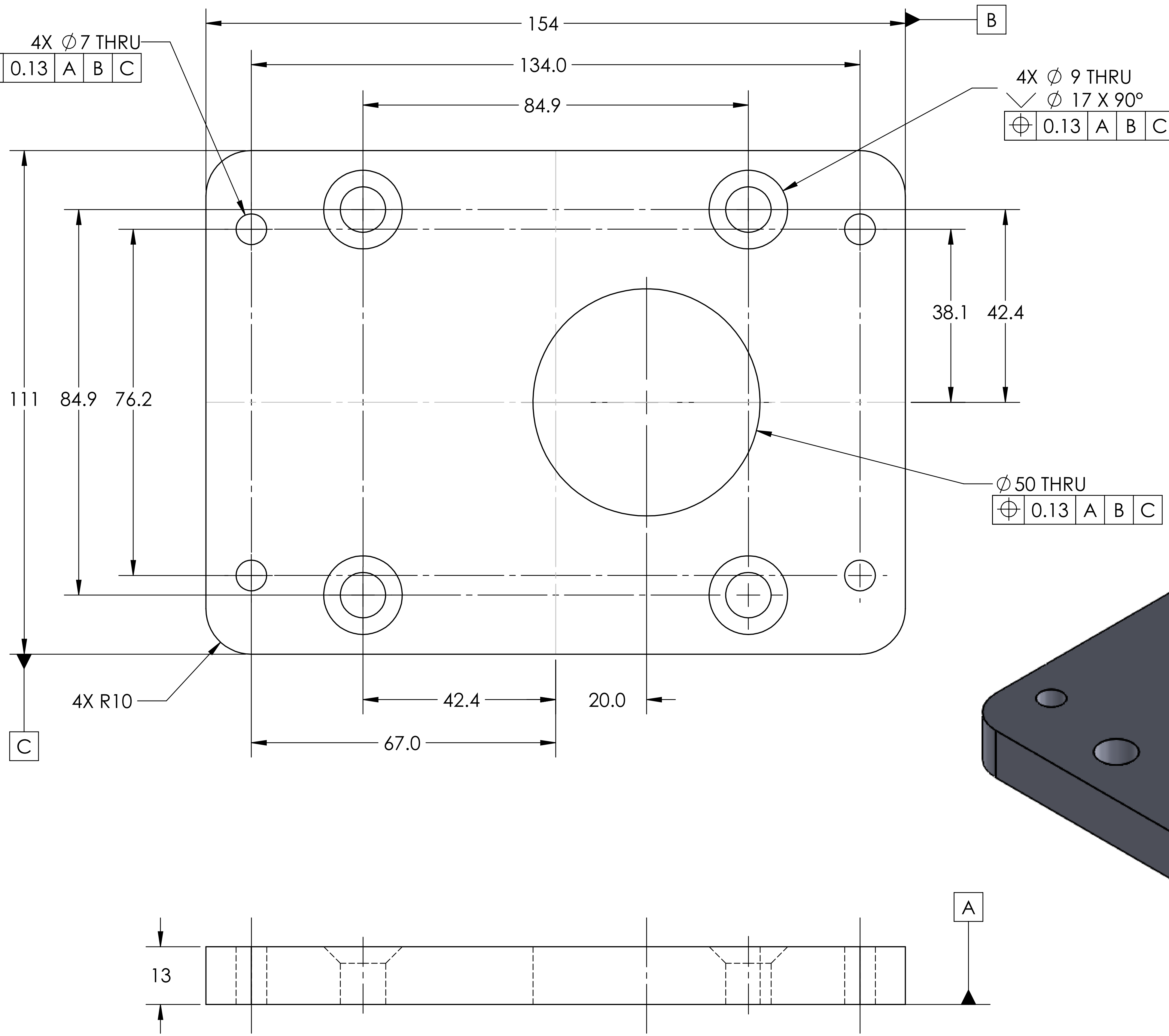




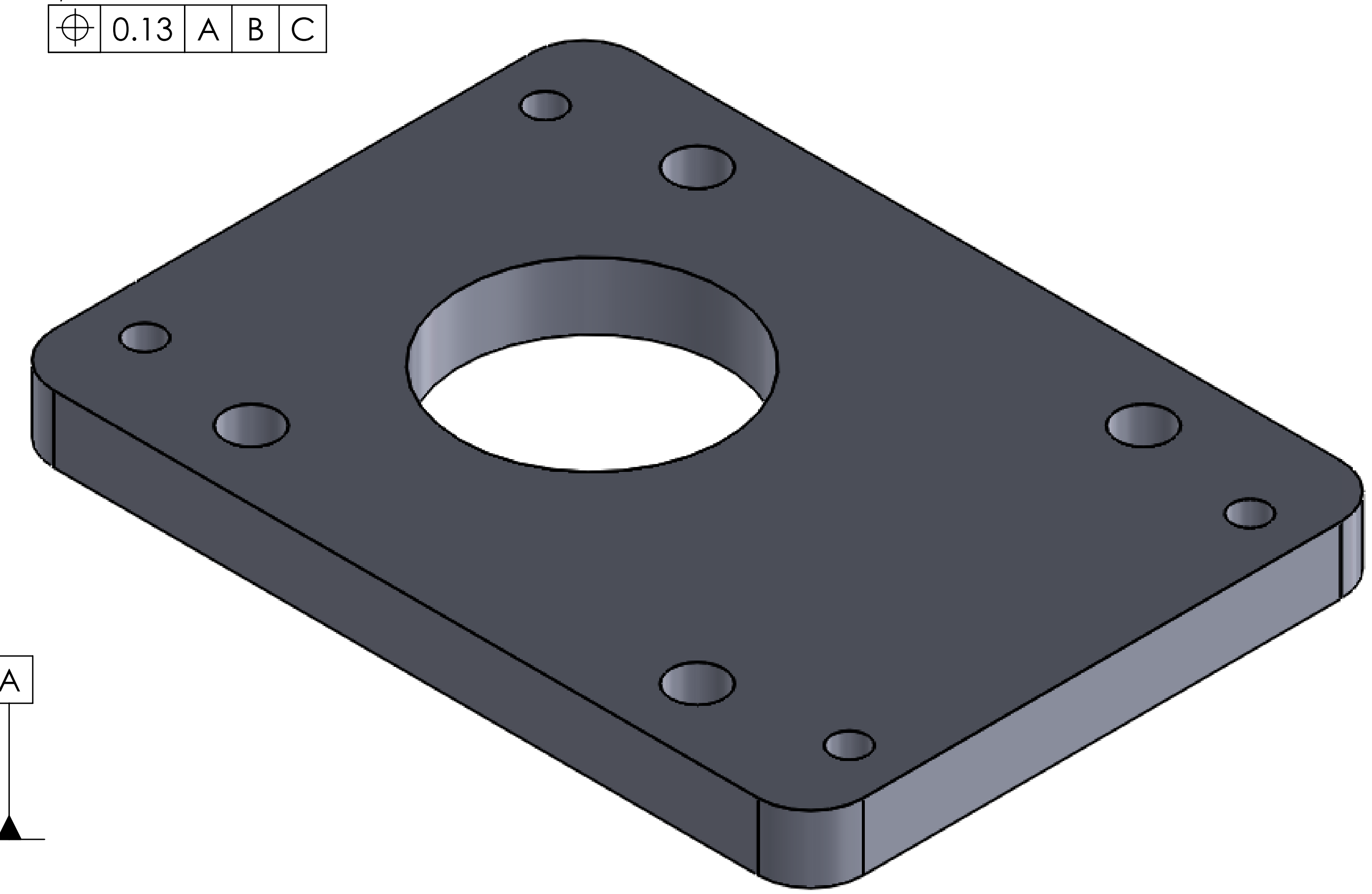


UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01

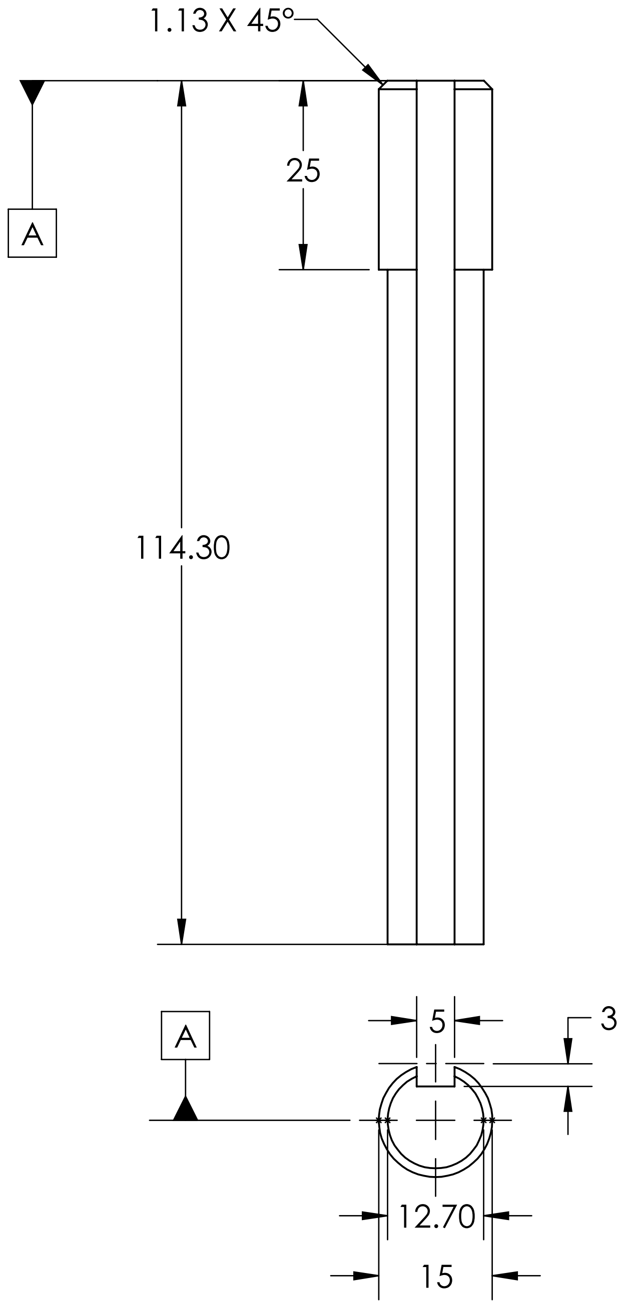
Cal Poly Mechanical Engineering	Lab Section:	Title: Plate Mounting Bracket		Drwn. By: Matthew Carlson
ME 430 - SPRING 2020	Dwg. #: 403	Date: 06/03/2020	Scale: 1:1	Chkd. By: Jackie Paik



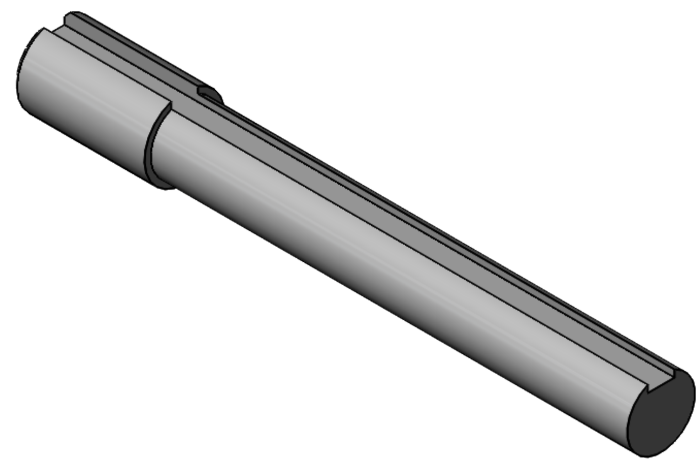
UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS ARE IN MM
2. MATERIAL: Polycarbonate Filament
3. TOLERANCE
X+/- .5
X.x+/-.2



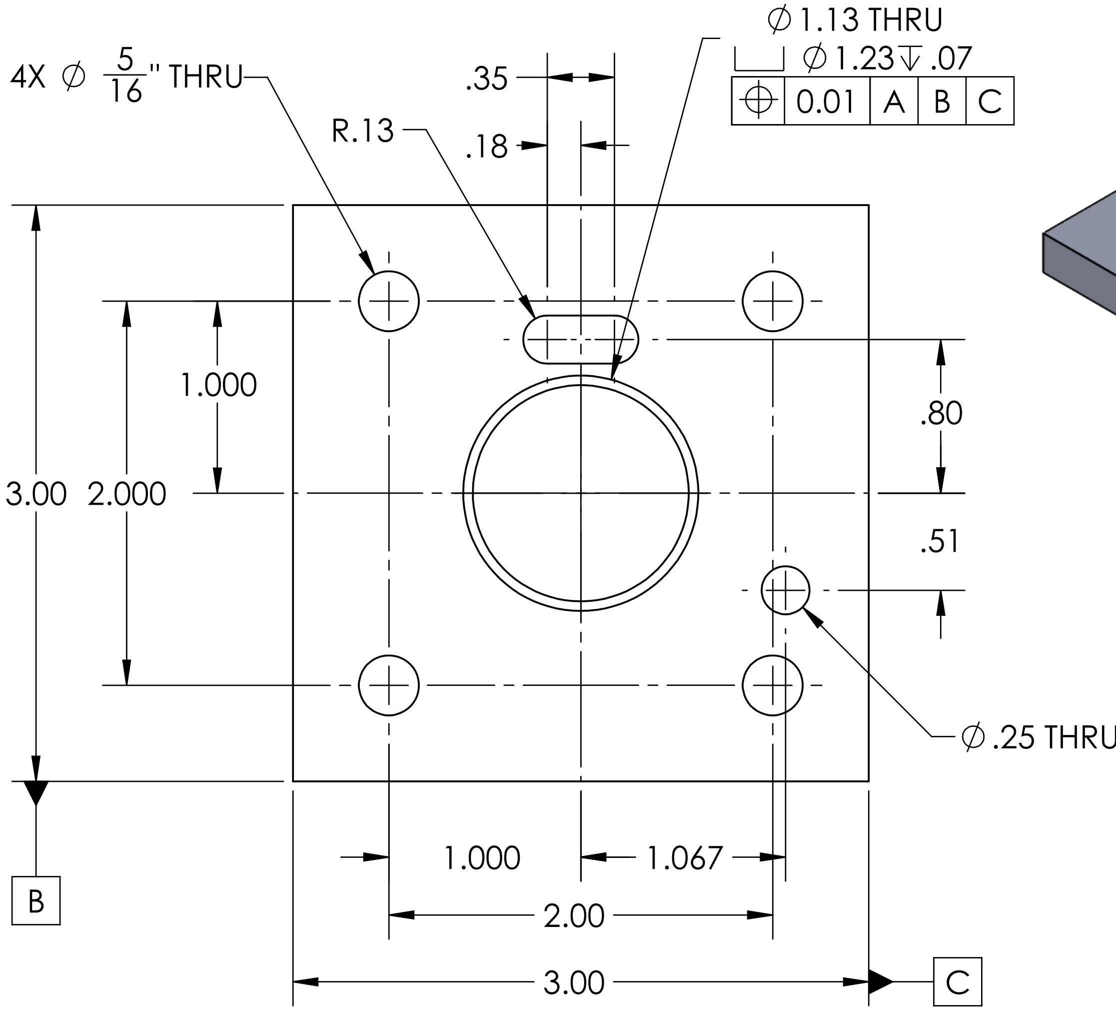
Cal Poly Mechanical Engineering	Lab Section:	Title: Primary Motor Mounting Plate		Drwn. By: MATTHEW CARLSON
ME 430 - SPRING 2020	Dwg. #: 914-5	Date: 06/03/20	Scale: 1:1	Chkd. By: Jackie Paik

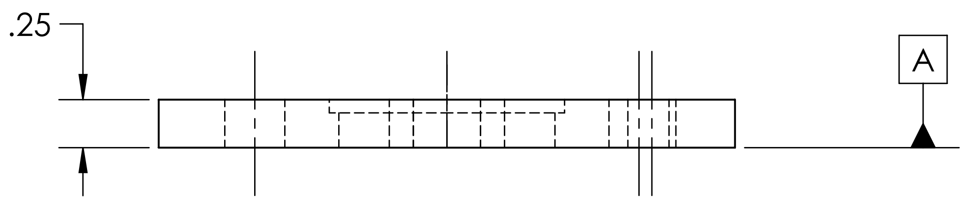


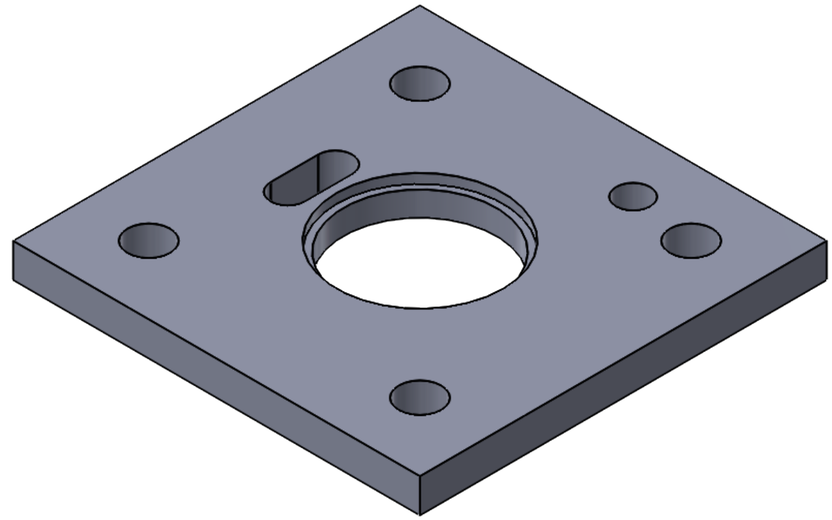
UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS IN MM
2. TOLERANCES
   X +/- .5
   X.X +/- .2
   +/- .1°
3. This drawing is a combination of 106 and 109



Cal Poly Mechanical Engineering ME 430 - SPRING 2020	Lab Section: Dwg. #: 106 & 109	Title: Primary Shaft Date: 06/03/2020 / Scale: 1:1	Drwn. By: Matthew Carlson Chkd. By: Jackie Paik





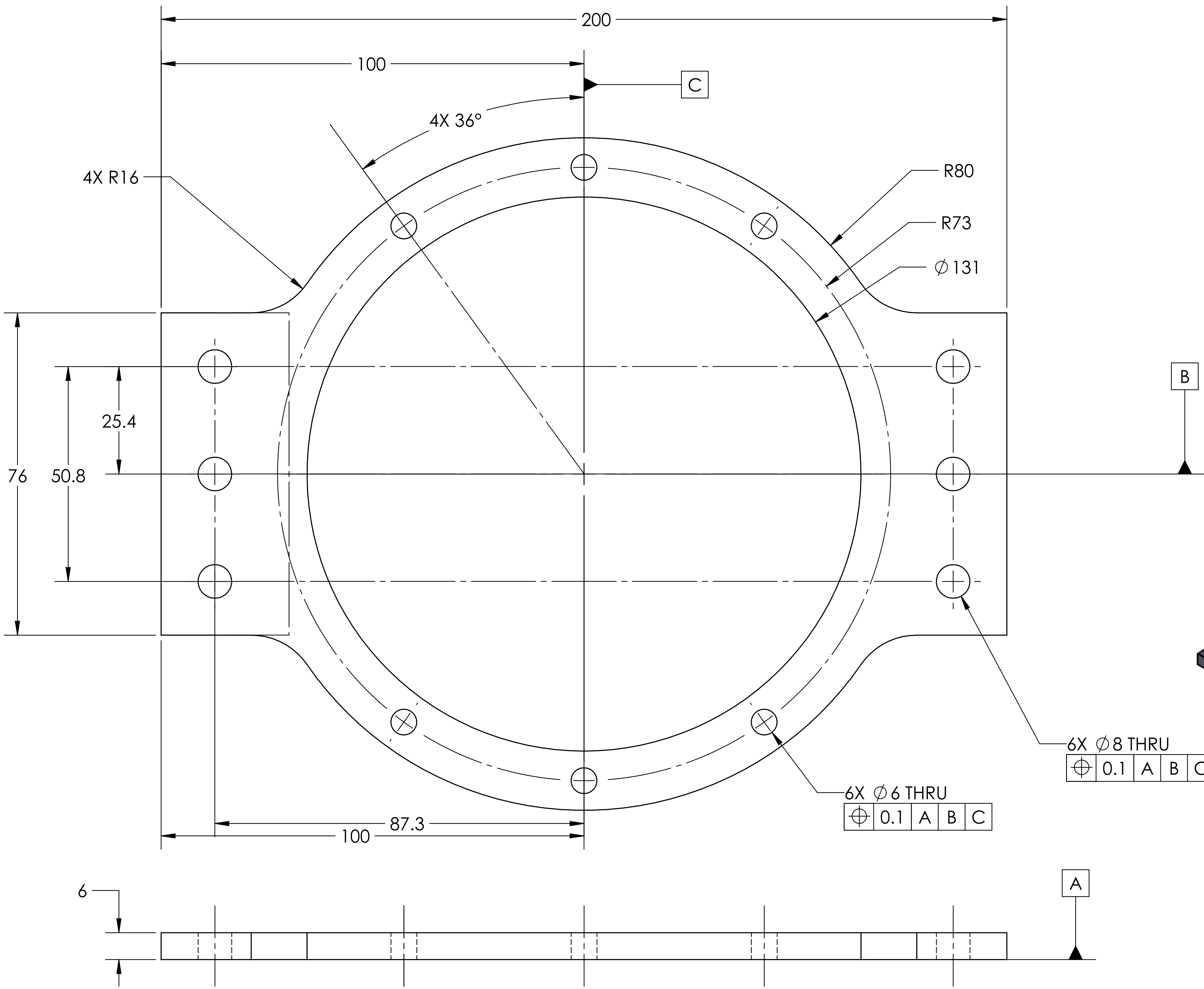


UNLESS OTHERWISE SPECIFIED

1. ALL DIMENSIONS ARE IN INCHES
2. MATERIAL: ALUMINUM 6061
3. INTERPRET PER ASME Y14.5 2018
4. TOLERANCES
5. X.XX +/- .02
6. X.XXX +/-.01

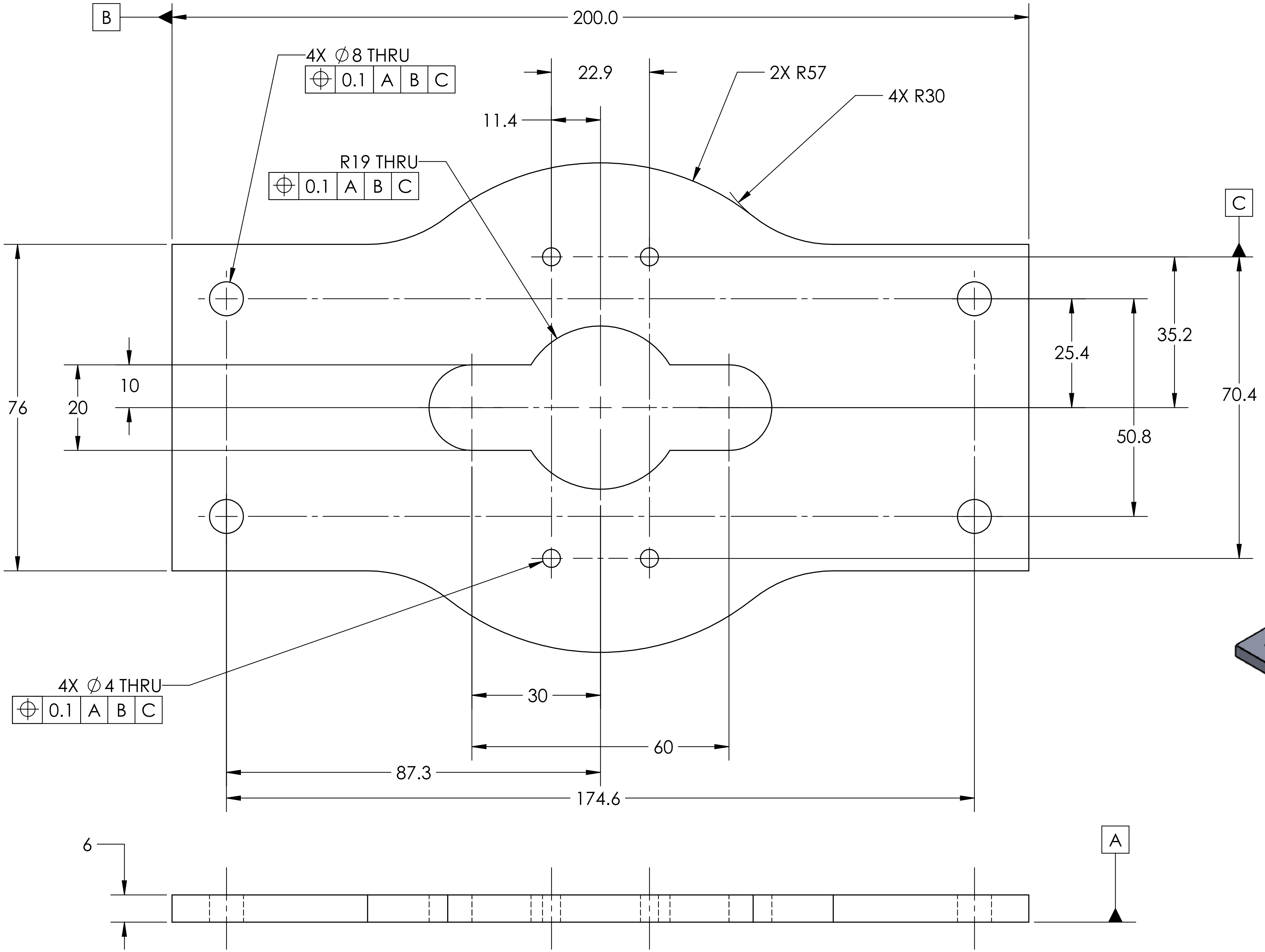
Cal Poly Mechanical Engineering	Lab Section:	Title: Slip Ring Bearing Mounting Plate	Drwn. By: Matthew Carlson
ME 430 - SPRING 2020	Dwg. #: 902-3	Date: 06/03/2020 · Scale: 1:1	Chkd. By: Jackie Paik

UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS IN MM
2. MATERIAL: ALUMINUM 6061
3. TOLERANCE
X+/- .5
X.X+/-.2
+/-.1°

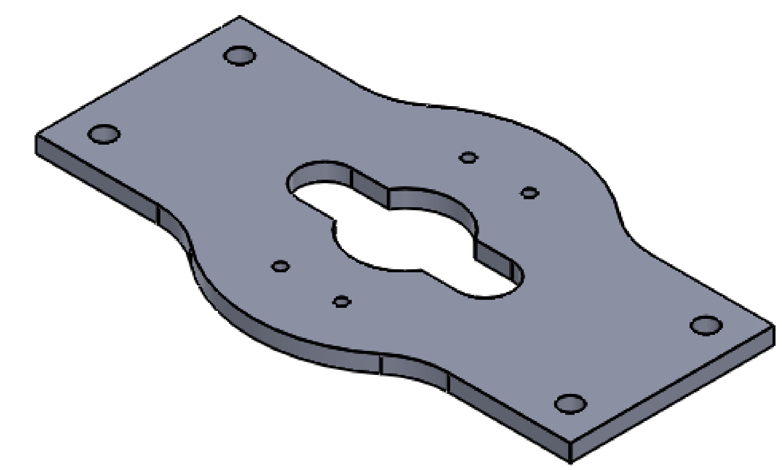


Scale 1:2

Cal Poly Mechanical Engineering	Lab Section:	Title: Turn Table Mounting Plate Base	Drwn. By: MATTHEW CARLSON
ME 430 - SPRING 2020	Dwg. #: 902-5	Date: 06/03/20 — Scale: 1:1	Chkd. By: Jackie Paik



UNLESS OTHERWISE SPECIFIED
1. ALL DIMENSIONS IN MM
2. MATERIAL: ALUMINUM 6061
3. TOLERANCE
X+/- .5
X.x+/-.2



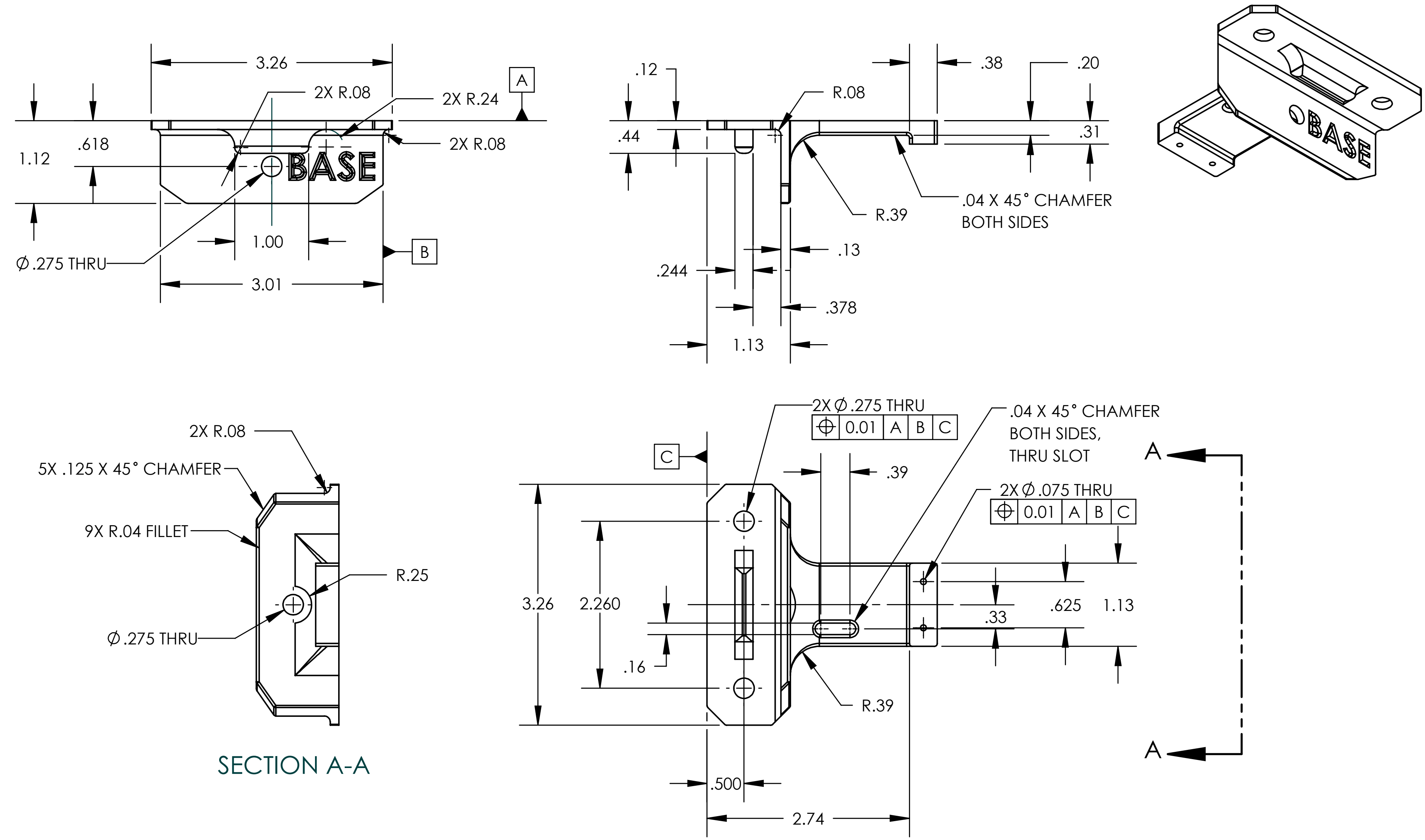
Scale 1:2

Cal Poly Mechanical Engineering	Lab Section:	Title: Turn Table Mounting Plate Fork		Drwn. By: MATTHEW CARLSON
ME 430 - SPRING 2020	Dwg. #: 902-6	Date: 06/03/20	Scale: 1:1	Chkd. By: Jackie Paik

NOTES: UNLESS OTHERWISE SPECIFIED

1. MATERIAL: POLYMAX POLYCARBONATE
2. ALL HOLES REAMED TO SIZE WITH HANDHELD DRILL
3. ALL UNDIMENSIONED FILLETS R.04
4. ALL UNDIMENSIONED CORNER CHAMFERS .25 X 45°

REVISIONS				
REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	5/6/2020	TVDB
B		CHANGED TOLERANCES, ALTERED DIMS, ADDED GD&T	6/3/2020	TVDB
C		ENLARGED ISOMETRIC VIEW	6/3/2020	TVDB



INSPIRED FLIGHT
SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES

BREAK EDGES	DECIMALS	FINISHED SURFACES
.004/.012	.XX ± .02 .XXX ± .01	.8 Ra
		ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

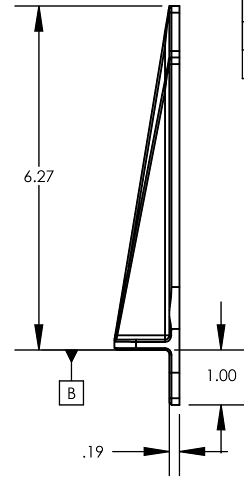
THIRD ANGLE PROJECTION

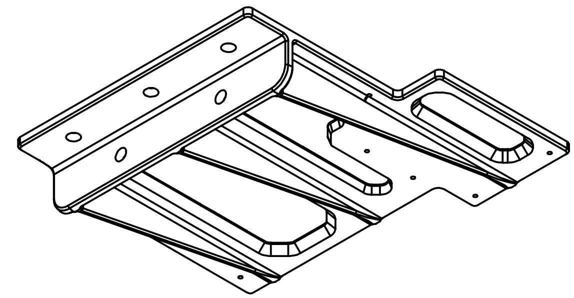
DRAWN: TVDB	DATE: 5/6/2020	BASE PROXIMITY SENSOR MOUNT		
CHECK/ENG:	DATE:			
APP:	DATE:	SIZE A	PART NO:	REV: C
REFERENCE FILENAME: Base Proximity Sensor Mount V1		SCALE: 1:2	DO NOT SCALE DRAWING	SHEET: 1 OF 1

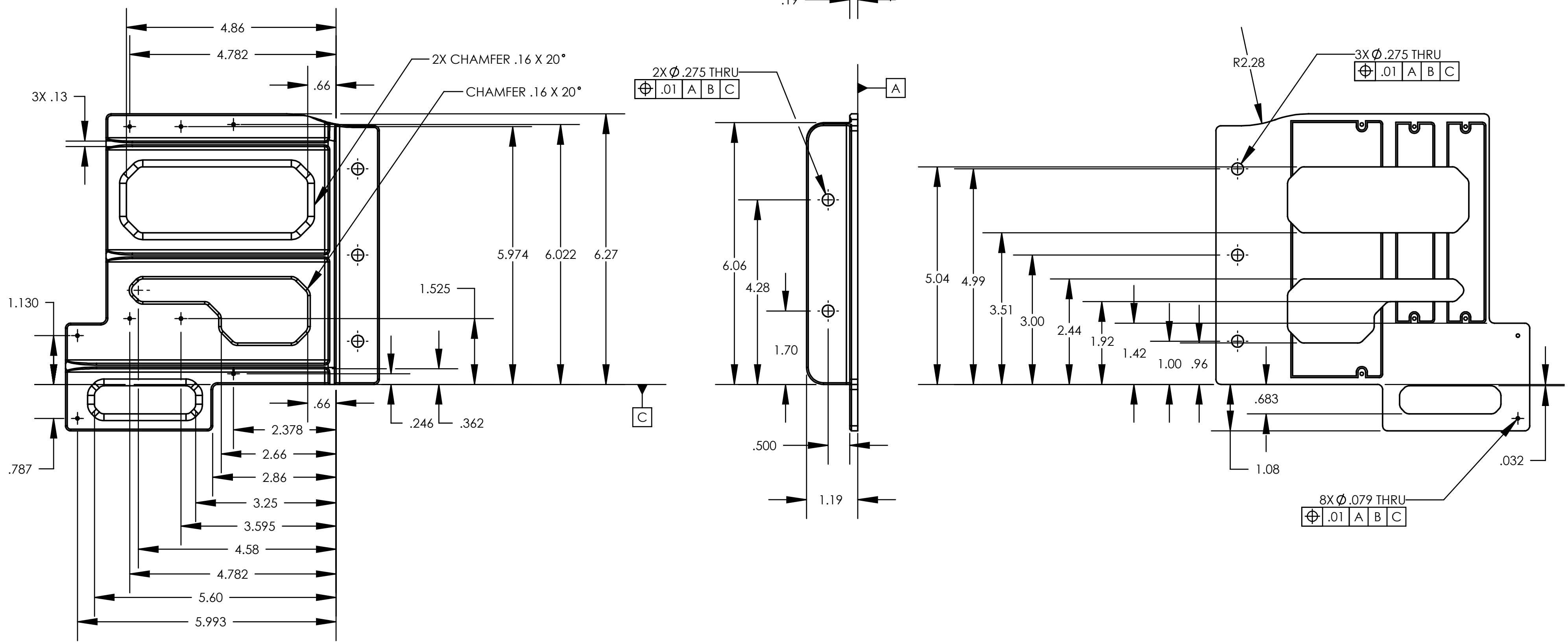
NOTES: UNLESS OTHERWISE SPECIFIED

1. MATERIAL: POLYMAX POLYCARBONATE
2. ALL HOLES REAMED TO SIZE WITH HANDHELD DRILL
3. ALL UNDIMENSIONED FILLETS R.04
4. ALL UNDIMENSIONED CORNER CHAMFERS .25 X 45°

REVISIONS				
REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	6/3/2020	TVDB
B		ENLARGED ISOMETRIC VIEW	6/3/2020	TVDB







INSPIRED FLIGHT
SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES

BREAK EDGES	DECIMALS	FINISHED SURFACES
.004/.012	.XX ± .02 .XXX ± .01	.8 Ra
		ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

THIRD ANGLE PROJECTION

DRAWN: TVDB	DATE: 6/3/20		
CHECK/ENG:	DATE:		
APP:	DATE:		
REFERENCE FILENAME: Driver Mounting Plate V1			

FORK PROXIMITY SENSOR MOUNT

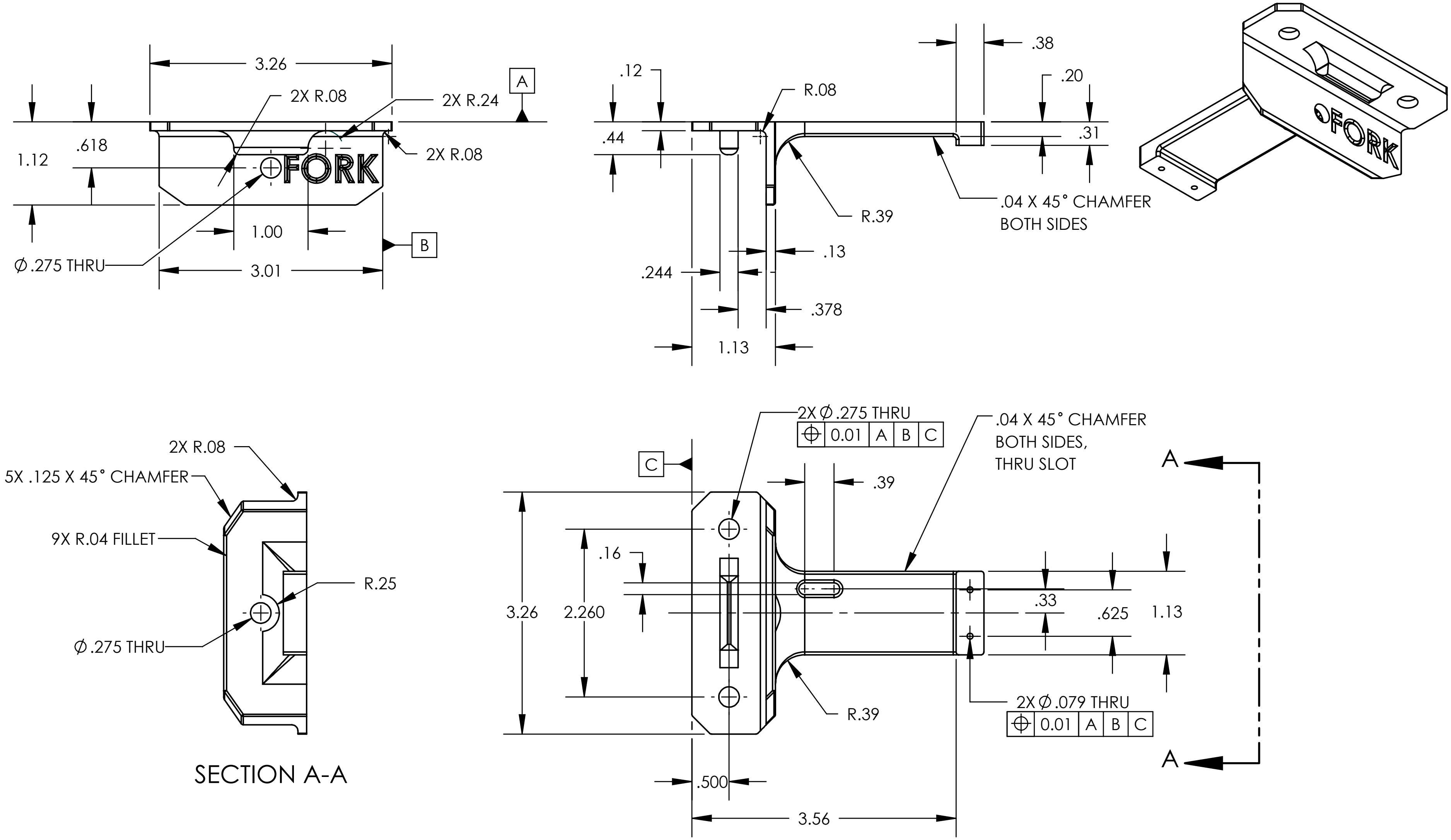
SIZE	PART NO:	REV:
A		B

SCALE:		SHEET:
1:3.5	DO NOT SCALE DRAWING	1 OF 1

NOTES: UNLESS OTHERWISE SPECIFIED

1. MATERIAL: POLYMAX POLYCARBONATE
2. ALL HOLES REAMED TO SIZE WITH HANDHELD DRILL
3. ALL UNDIMENSIONED FILLETS R.04
4. ALL UNDIMENSIONED CORNER CHAMFERS .25 X 45°

REVISIONS				
REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	5/6/2020	TVDB
B		CHANGED TOLERANCES, ALTERED DIMS, ADDED GD&T	6/3/2020	TVDB
C		ENLARGED ISOMETRIC VIEW	6/3/2020	TVDB



INSPIRED FLIGHT

SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES

BREAK EDGES .004/.012

DECIMALS .XX ± .02 .XXX ± .01

FINISHED SURFACES .8 Ra

ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

THIRD ANGLE PROJECTION

DRAWN: TVDB | DATE: 5/6/2020

CHECK/ENG: | DATE:

APP: | DATE:

REFERENCE FILENAME: Fork Proximity Sensor Mount V1

FORK PROXIMITY SENSOR MOUNT

SIZE A | PART NO: | REV: B

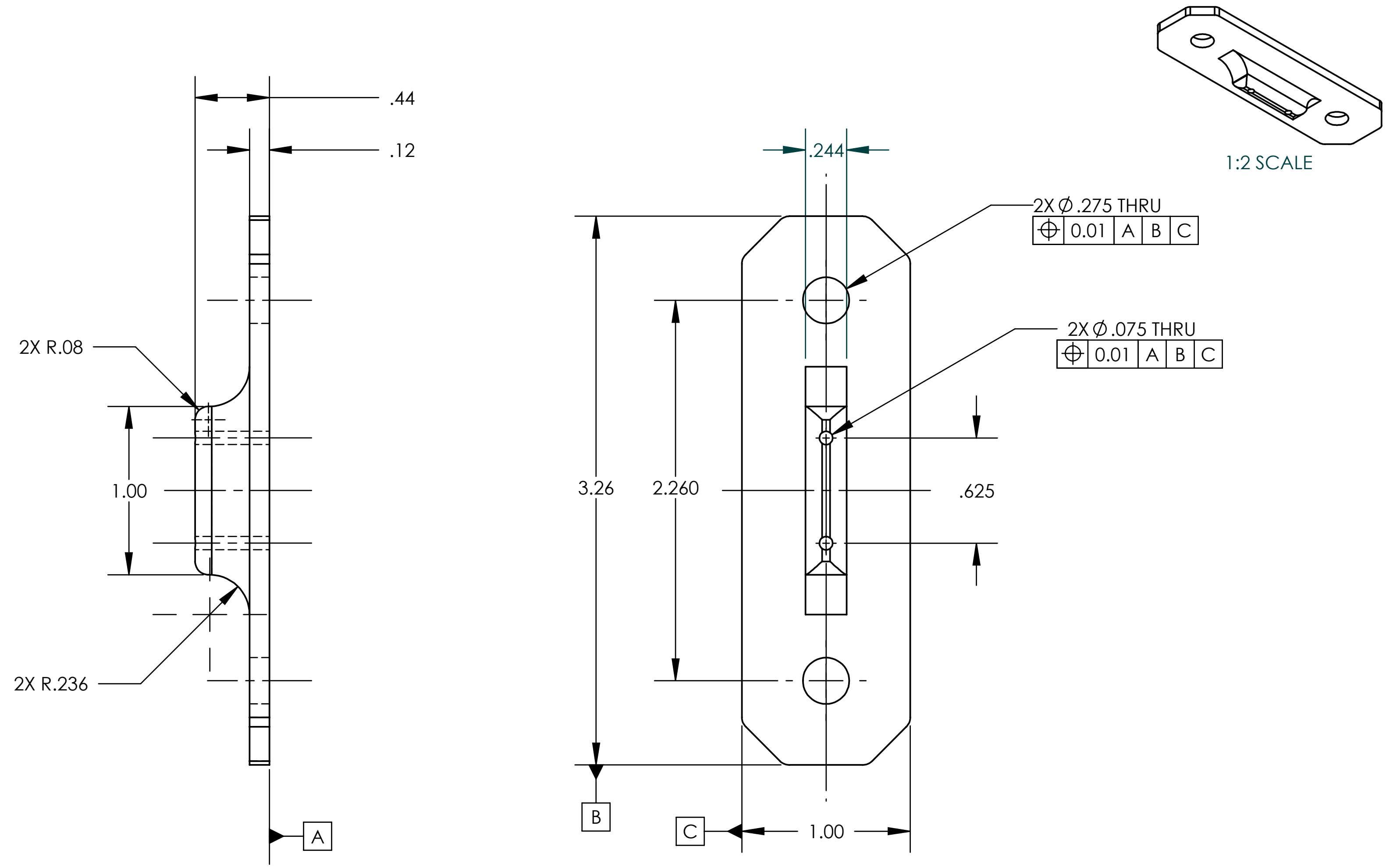
SCALE: 1:2 | DO NOT SCALE DRAWING | SHEET: 1 OF 1

NOTES: UNLESS OTHERWISE SPECIFIED

1. MATERIAL: POLYMAX POLYCARBONATE
2. ALL HOLES REAMED TO SIZE WITH HANDHELD DRILL
3. ALL UNDIMENSIONED FILLETS R.04
4. ALL UNDIMENSIONED CORNER CHAMFERS .25 X 45°

REVISIONS

REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	5/6/2020	TVDB
B		CHANGED TOLERANCES, ALTERED DIMS, ADDED GD&T	6/3/2020	TVDB
C		ENLARGED ISOMETRIC VIEW	6/3/2020	TVDB



INSPIRED FLIGHT
SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES

BREAK EDGES .004/.012
DECIMALS .XX ± .02 .XXX ± .01
FINISHED SURFACES .8 Ra
ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

THIRD ANGLE PROJECTION

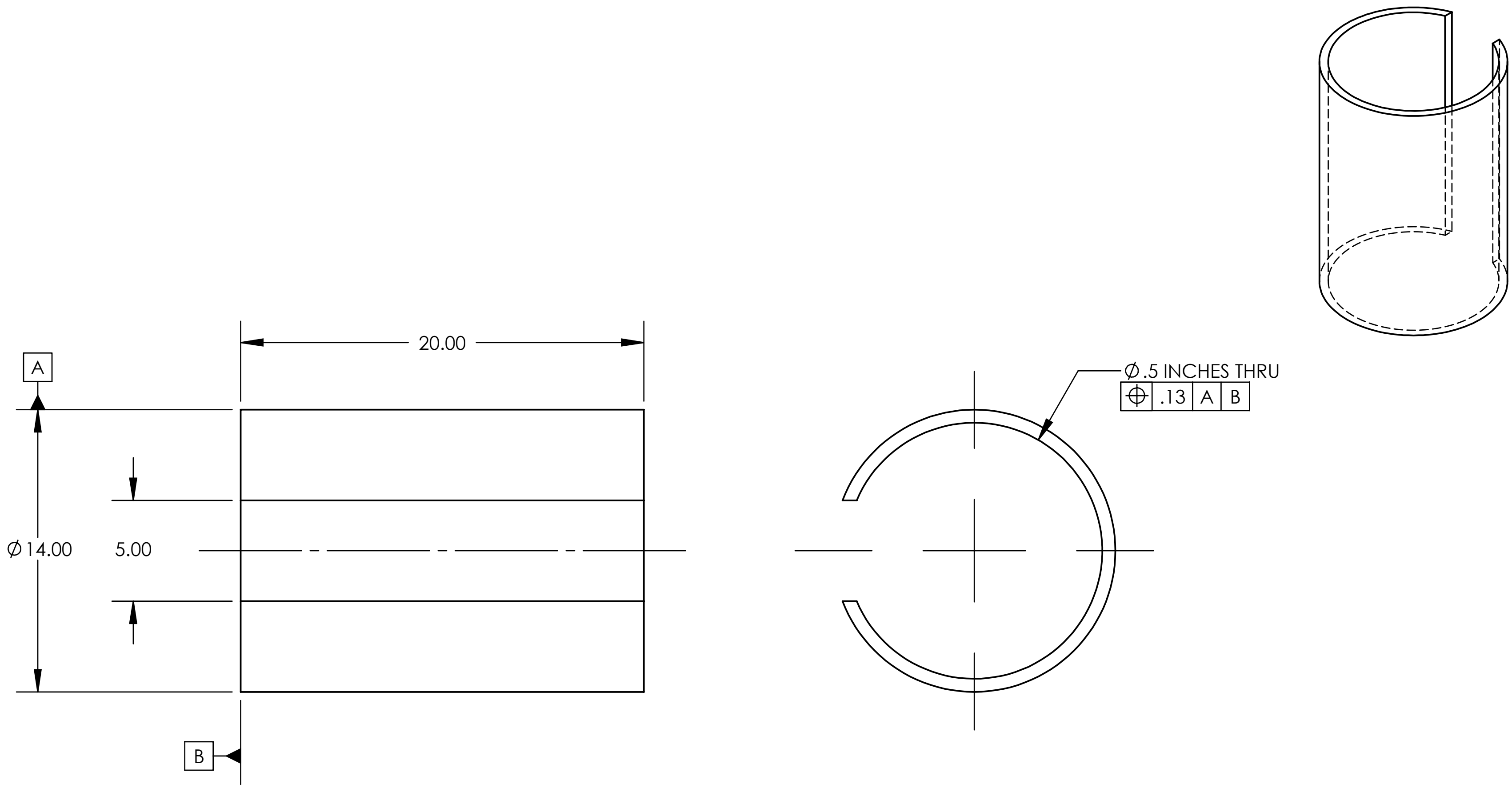
DRAWN: TVDB
DATE: 5/6/2020
CHECK/ENG:
DATE:
APP:
DATE:
REFERENCE FILENAME: Ring Proximity Sensor Mount V1

RING PROXIMITY SENSOR MOUNT

SIZE: A
PART NO:
REV: C
SCALE: 1:1
DO NOT SCALE DRAWING
SHEET: 1 OF 1

NOTE: MILL SLOT BETWEEN TWO PARALLELS ON MILL
MATERIAL: BRASS

REVISIONS				
REV	ECO	DESCRIPTION	DATE	DRAWN
A		INITIAL RELEASE	6/5/2020	TVDB



INSPIRED FLIGHT
SAN LUIS OBISPO, CA

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN MM

BREAK EDGES	DECIMALS	FINISHED SURFACES
.1/.3	.X ± .5 .XX ± .13	.8 Ra
		ANGLES ± 0.1 °

DIMENSIONS & TOLERANCES PER ASME Y14.5-2018

THIRD ANGLE PROJECTION

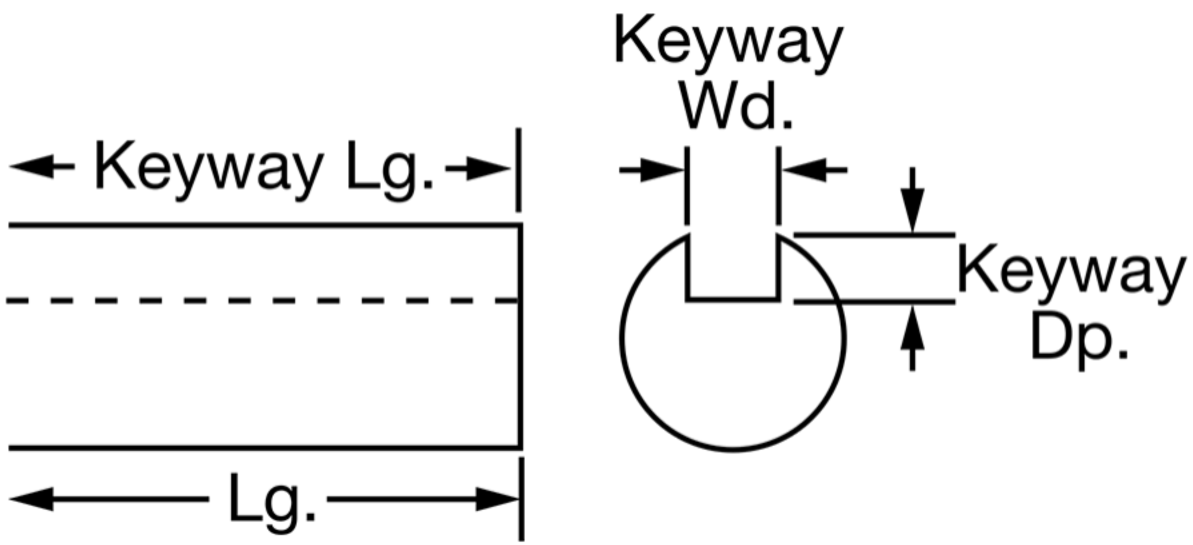
DRAWN: TVDB	DATE: 6/5/2020	BRASS ADAPTER SLEEVE	
CHECK/ENG:	DATE:		
APP:	DATE:	SIZE: A / PART NO:	REV: A
REFERENCE FILENAME: Brass Sleeve V1		SCALE: 1:1 / DO NOT SCALE DRAWING	SHEET: 1 OF 1

Is this page helpful?

1045 Carbon Steel Keyed Rotary Shaft

Fully Keyed, 14 mm Diameter, 200 mm Long

Fully Keyed

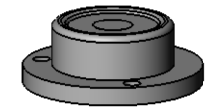


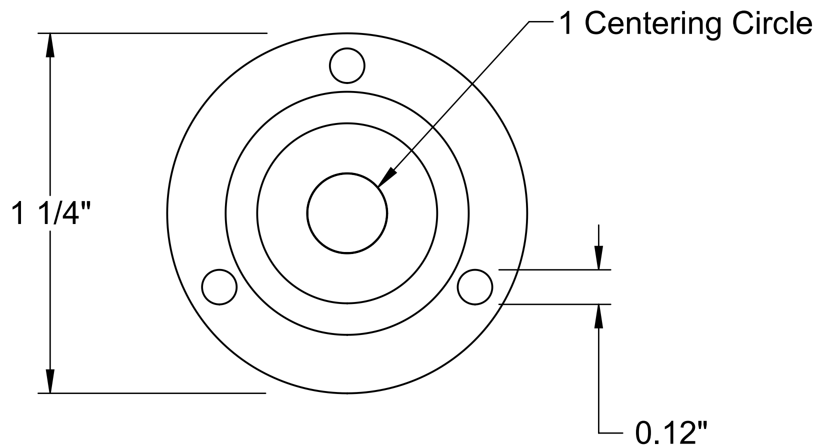
Each

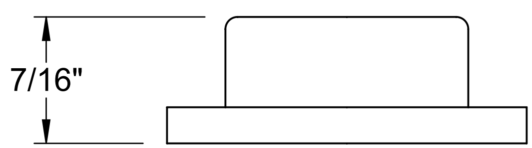
ADD TO ORDER

In stock
$15.84 Each
1439K26

For Motion Type	Rotary
End Type	Keyed × Keyed
Material	1045 Carbon Steel
Diameter	14mm
Diameter Tolerance	-0.0635mm to -0.0127mm
Length	200mm
Length Tolerance	-0.795mm to 0.795mm
Keyway	
Length	200mm
Width	5mm
Depth	3mm
ANSI Keys Included	No
Straightness Tolerance	0.33 mm per 300 mm
Edge Type	Chamfered
Hardness Rating	Medium
Hardness	Rockwell B95
Yield Strength	75,000 psi
Mechanical Finish	Turned, Precision Ground, Polished
System of Measurement	Metric





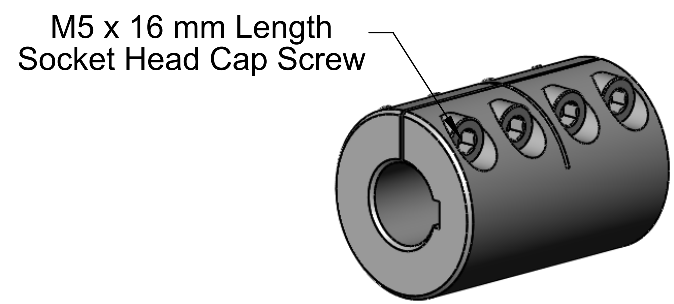


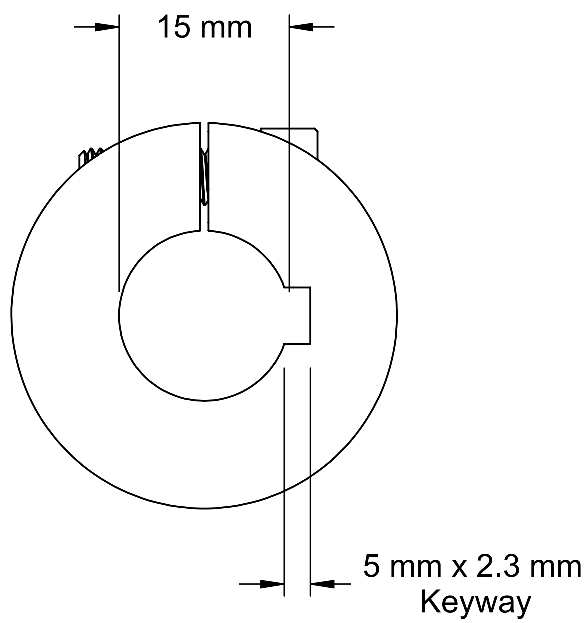
McMASTER-CARR® CAD

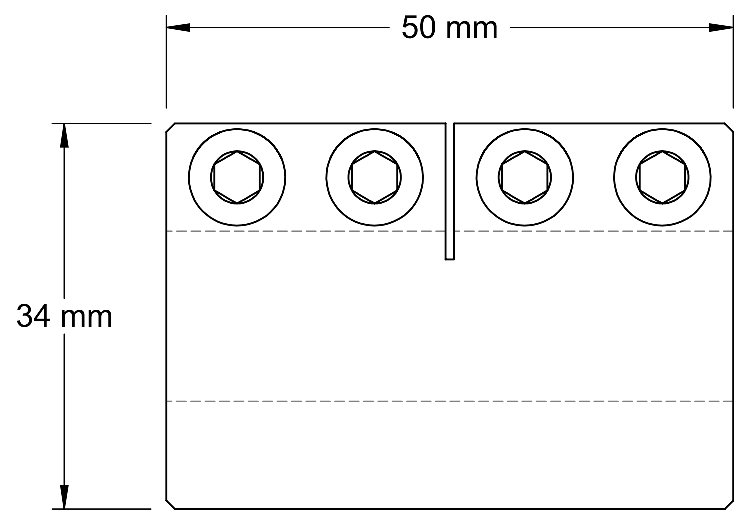
PART NUMBER **2198A85**

http://www.mcmaster.com

Bull's Eye
Level





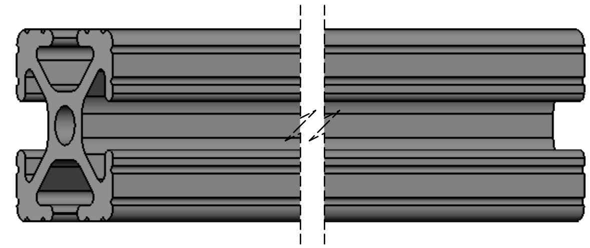


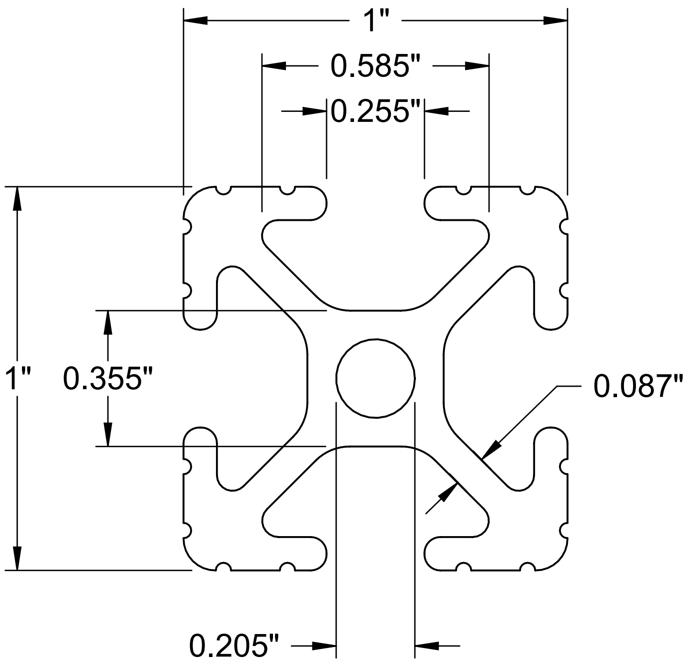
McMASTER-CARR® CAD

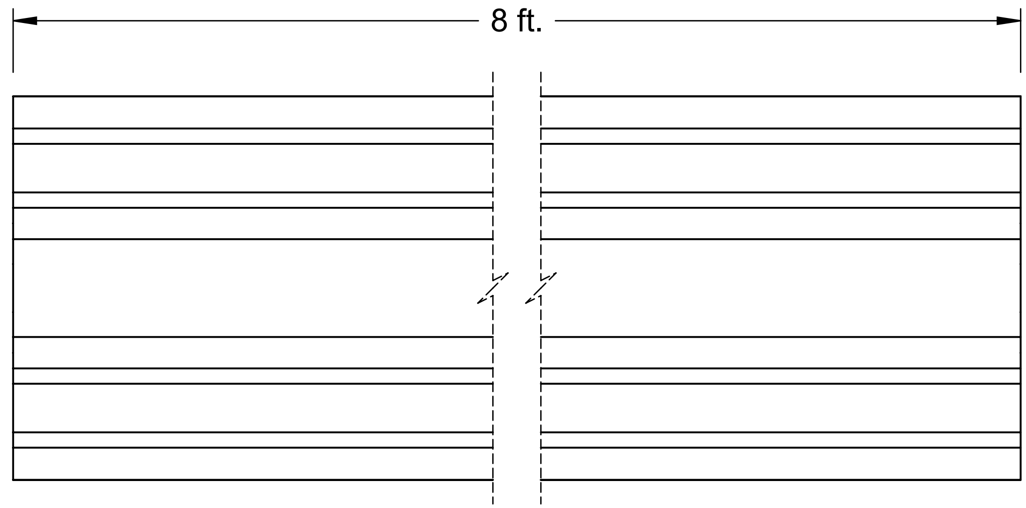
http://www.mcmaster.com

PART NUMBER **2469K75**

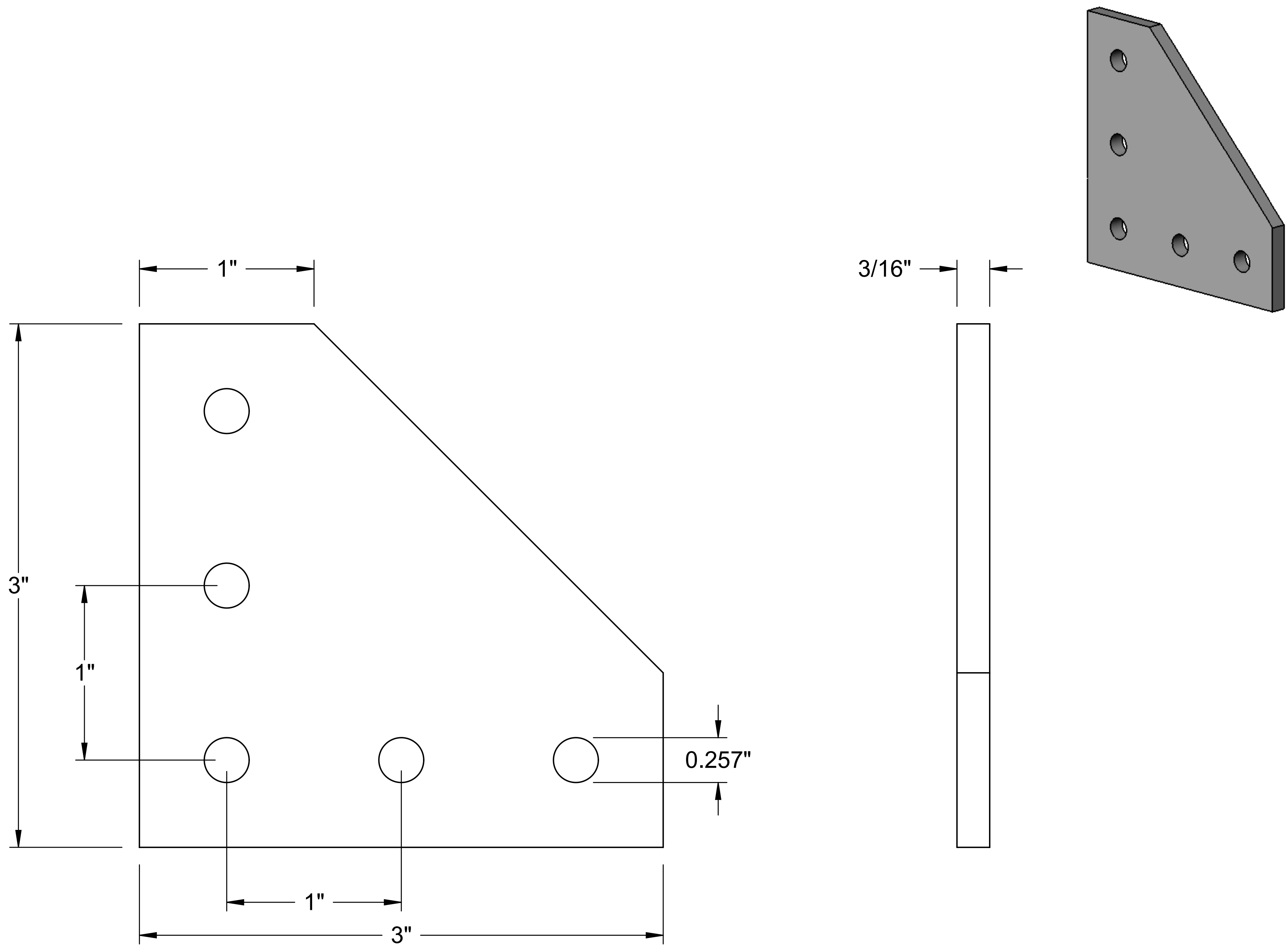
One-Piece Clamp-On Rigid Shaft Coupling







McMASTER-CARR® CAD	PART NUMBER	47065T101
http://www.mcmaster.com 	Aluminum T-Slotted Framing	

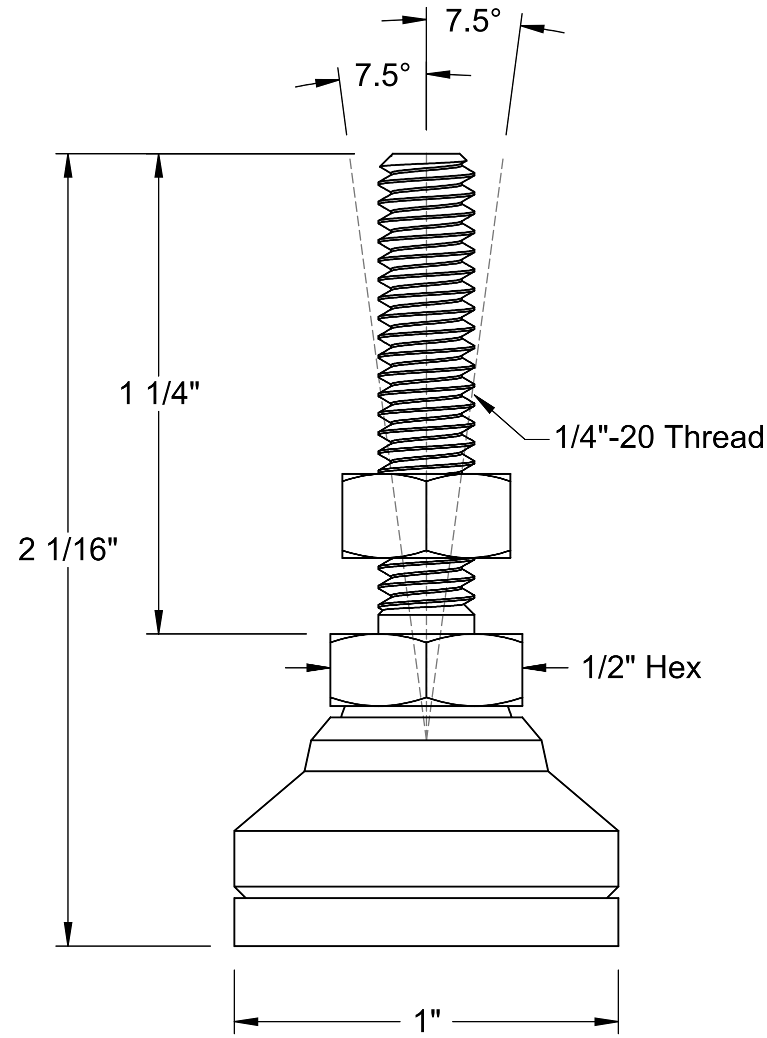


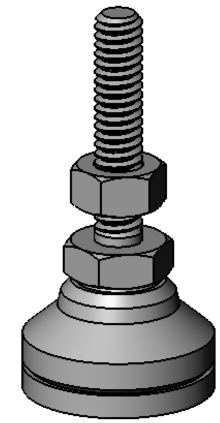
McMASTER-CARR® CAD

PART NUMBER **47065T267**

http://www.mcmaster.com

90° Plate for Aluminum T-Slotted Framing



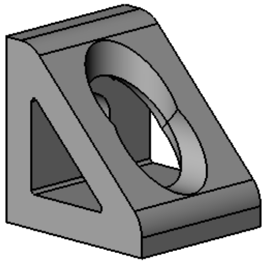


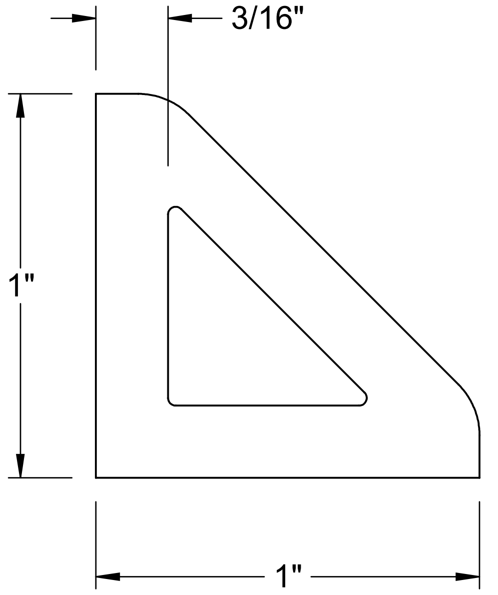
McMASTER-CARR® CAD

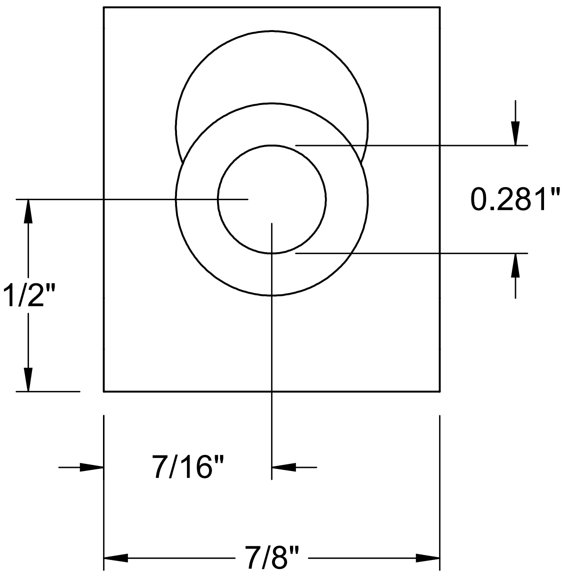
PART NUMBER **47065T341**

http://www.mcmaster.com

Leveling Mounting Foot
for T-Slotted Framing





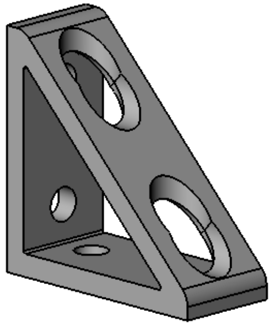


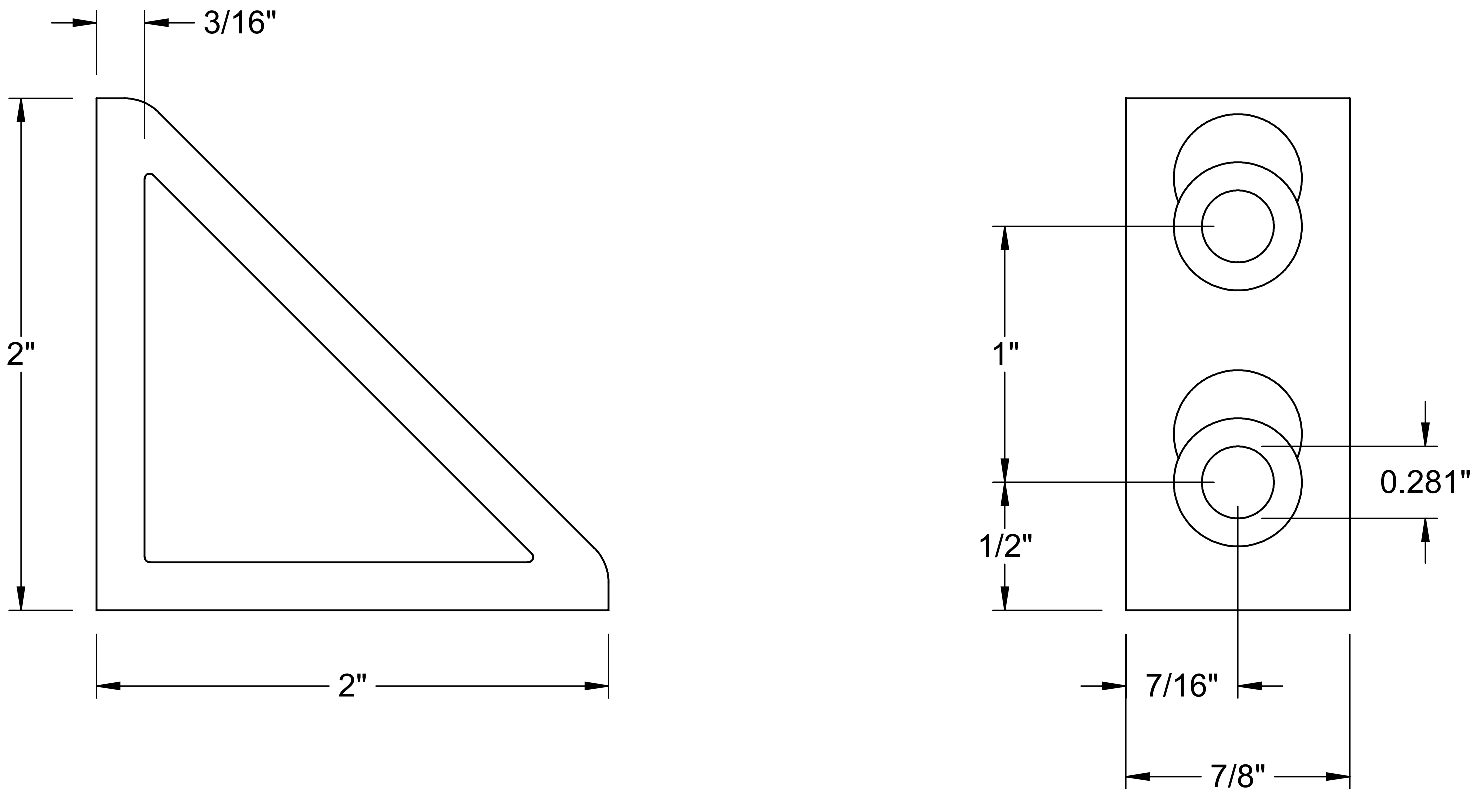
McMASTER-CARR® CAD

PART NUMBER 47065T663

http://www.mcmaster.com

Corner Bracket for
T-Slotted Framing



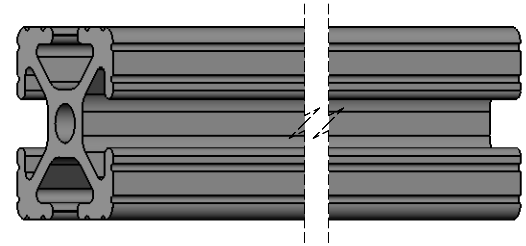


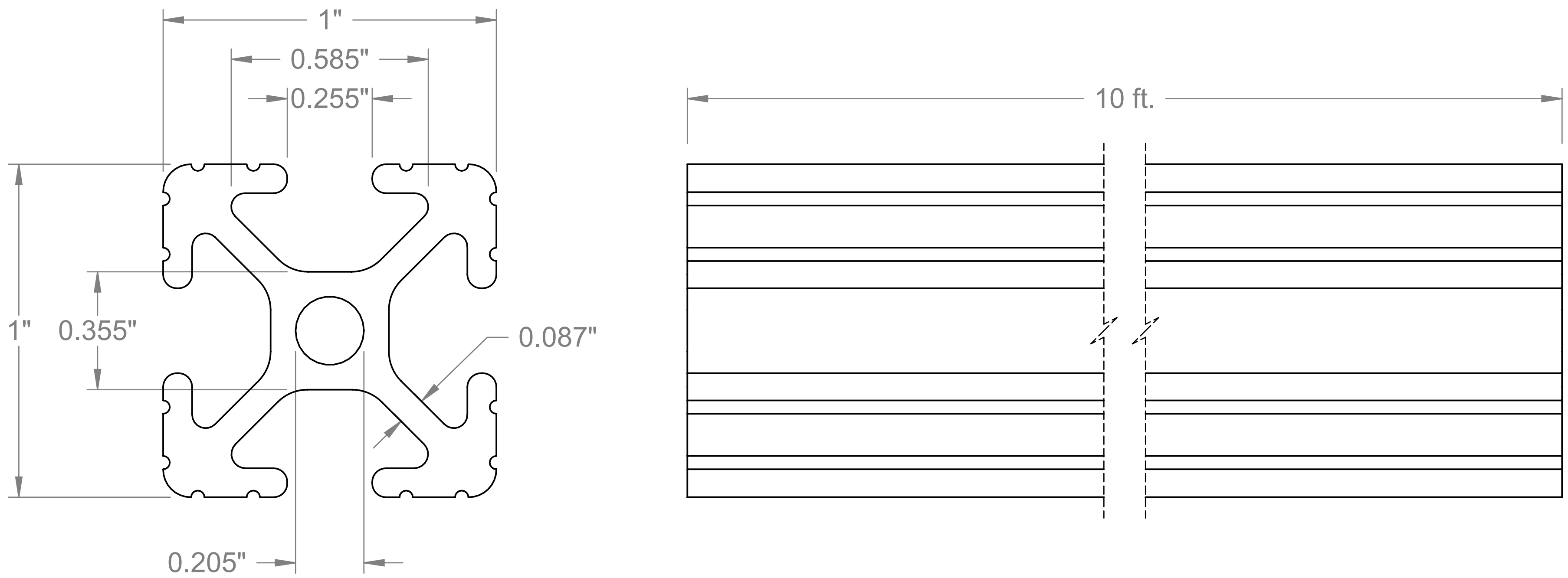
McMASTER-CARR® CAD

PART NUMBER 47065T736

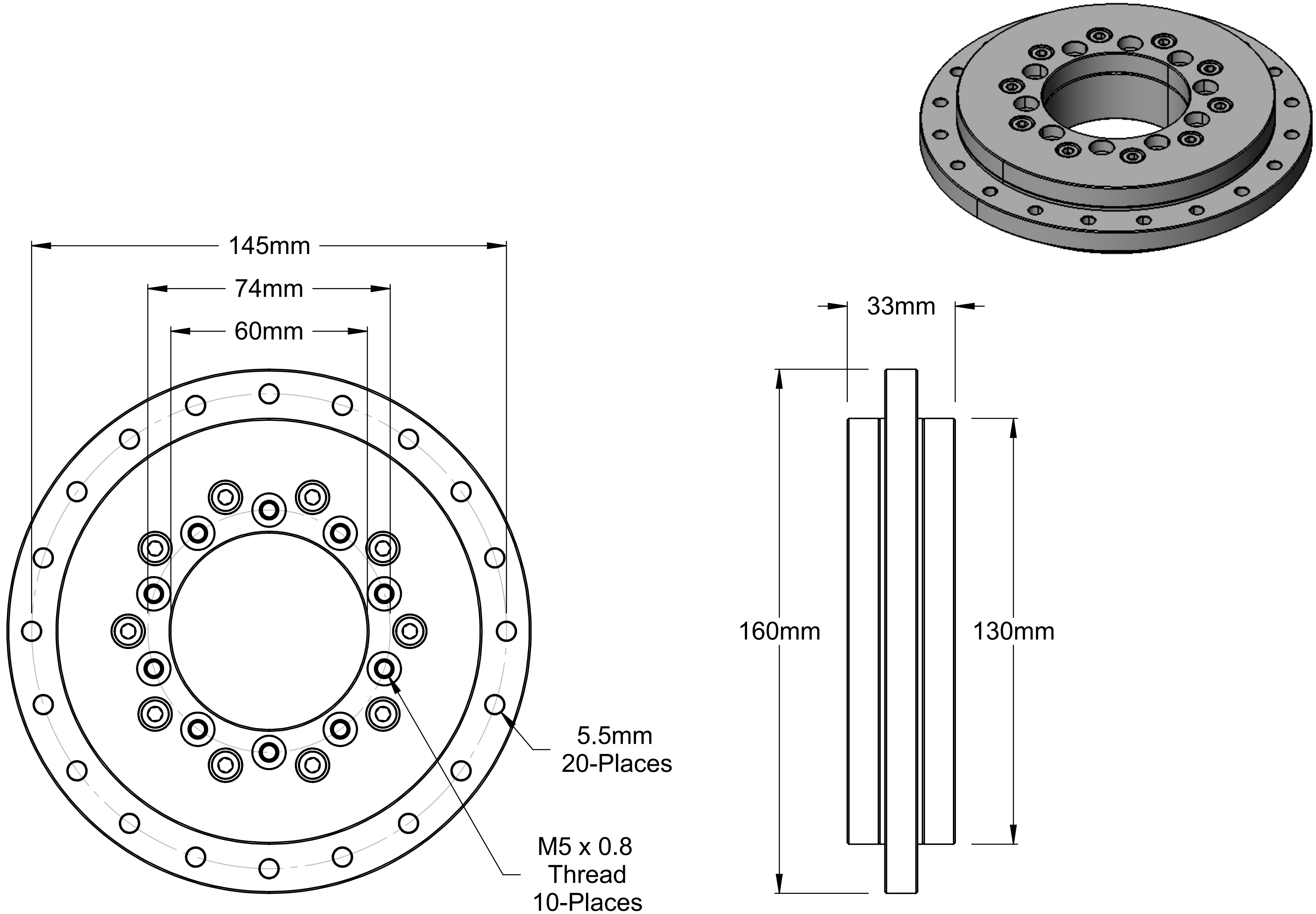
http://www.mcmaster.com

Corner Bracket for T-Slotted Framing

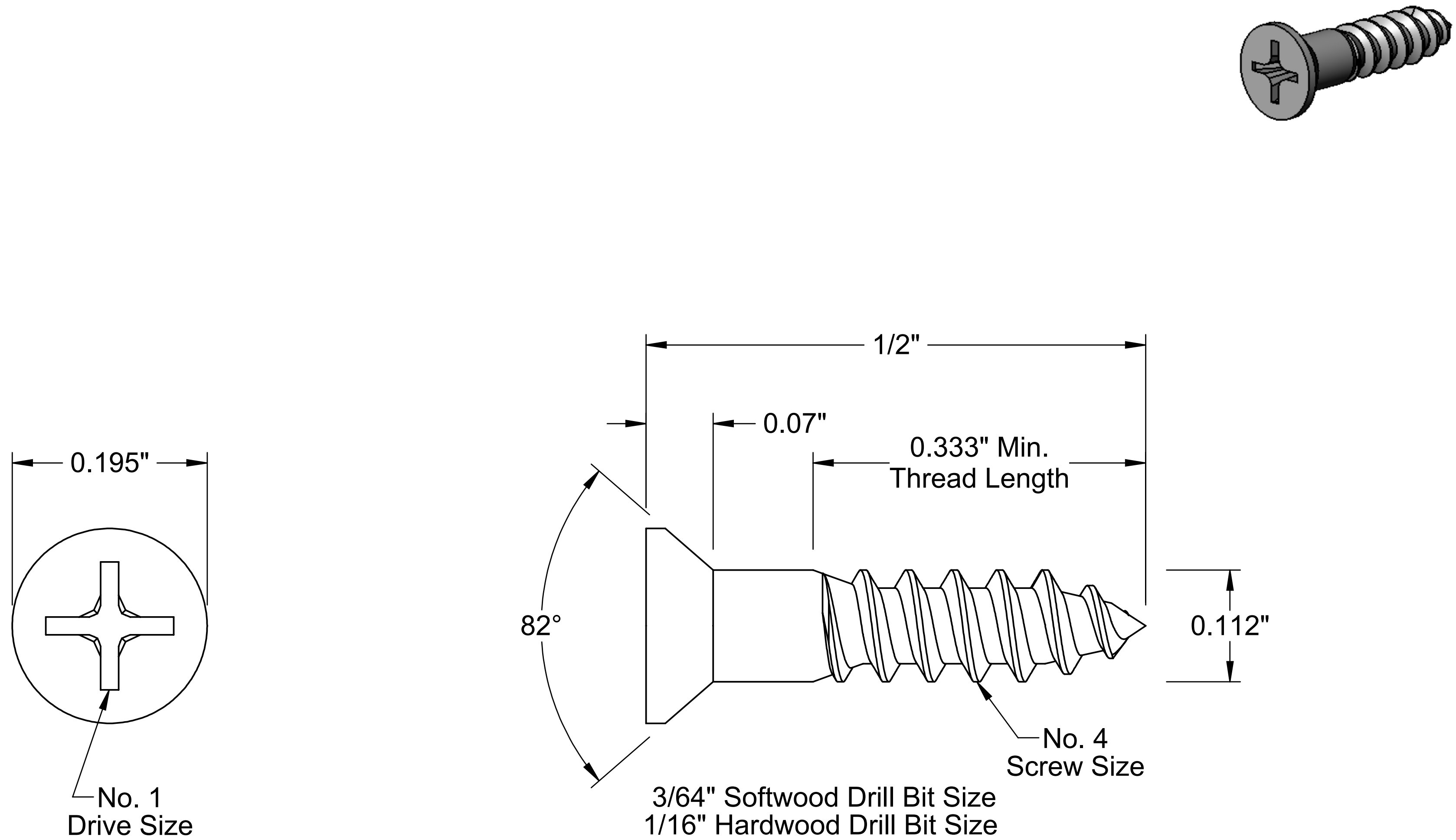




MCMASTER-CARR® CAD	PART NUMBER 47065T977
http://www.mcmaster.com © 2017 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Aluminum T-Slotted Framing



McMASTER-CARR® CAD	PART NUMBER	8700K2
http://www.mcmaster.com Information in this drawing is provided for reference only.	Sleeve-Bearing Turntable	

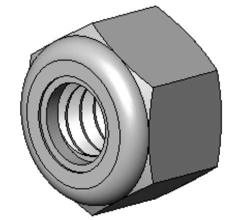


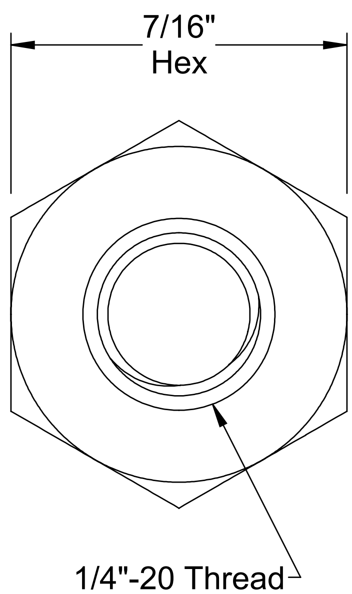
McMASTER-CARR® CAD

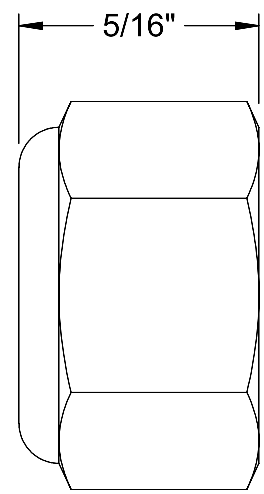
http://www.mcmaster.com

PART NUMBER **90031A110**

Steel Phillips
Flat Head Screw for Wood





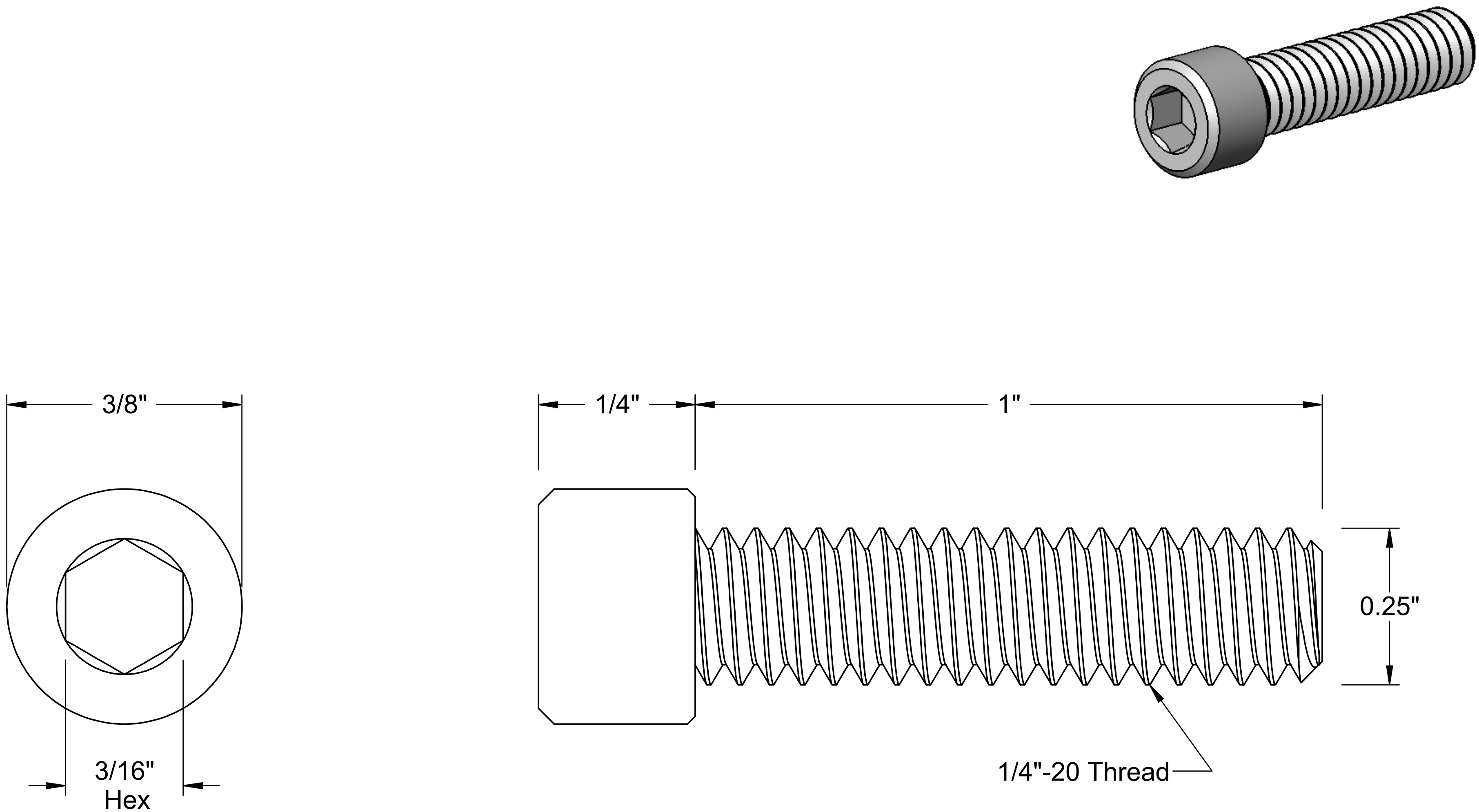


McMASTER-CARR® CAD

http://www.mcmaster.com

PART NUMBER **90630A110**

High-Strength Steel
Nylon-Insert Locknut

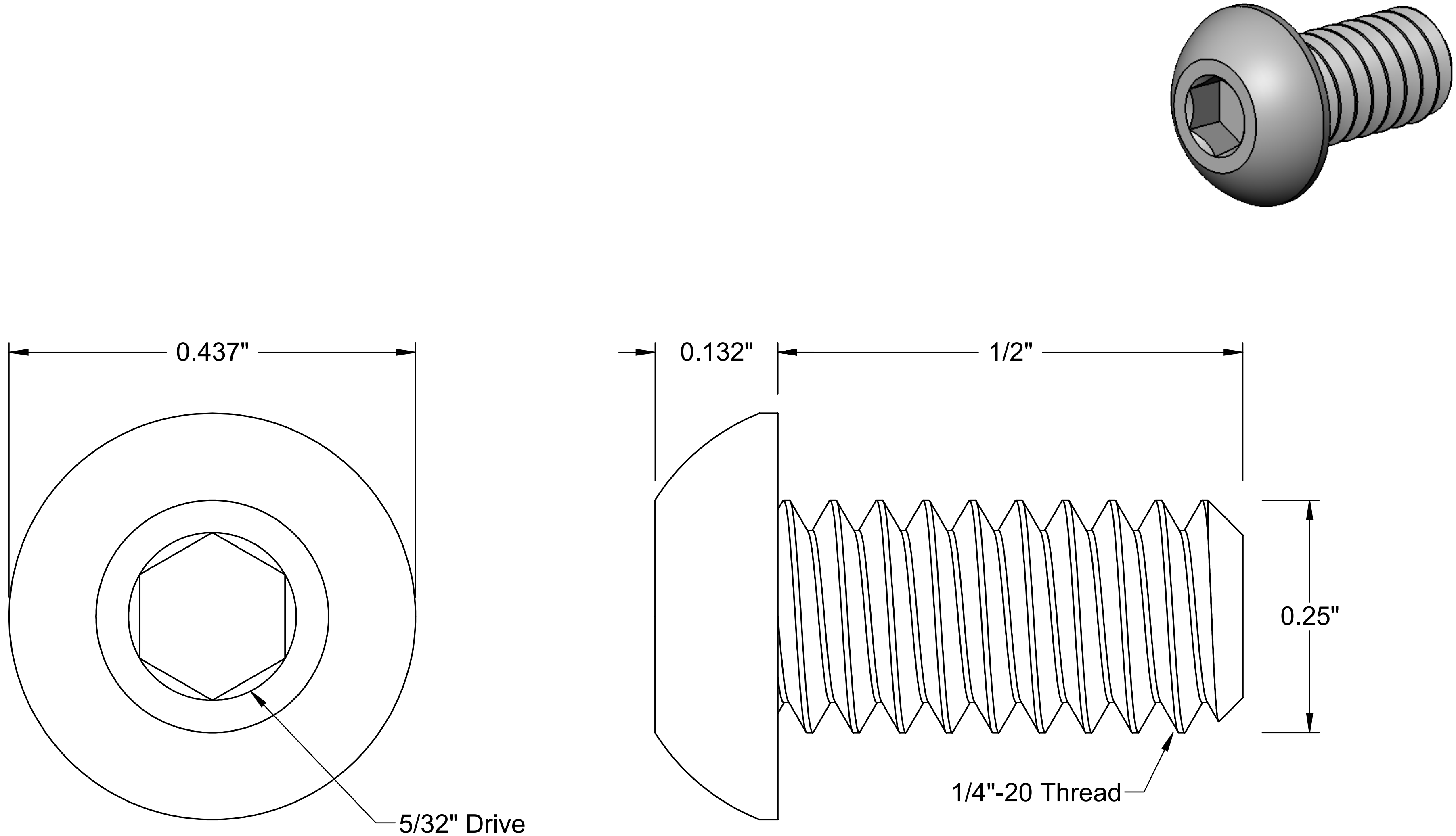


McMASTER-CARR® CAD

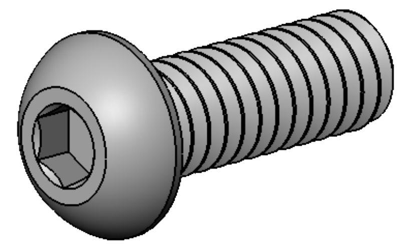
PART NUMBER **91251A542**

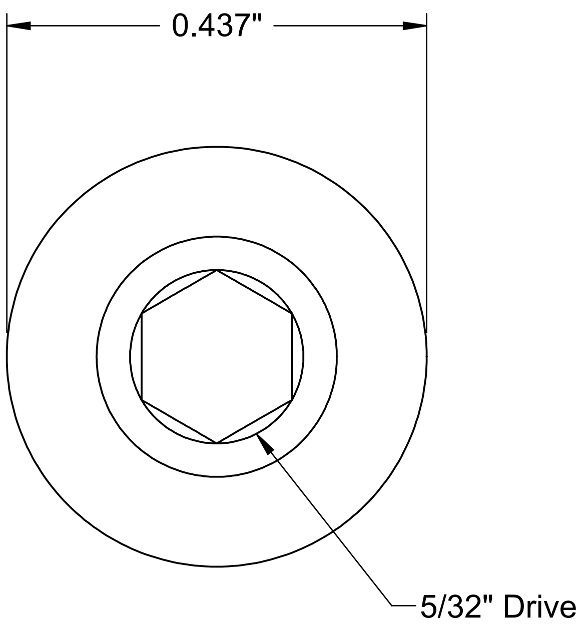
http://www.mcmaster.com

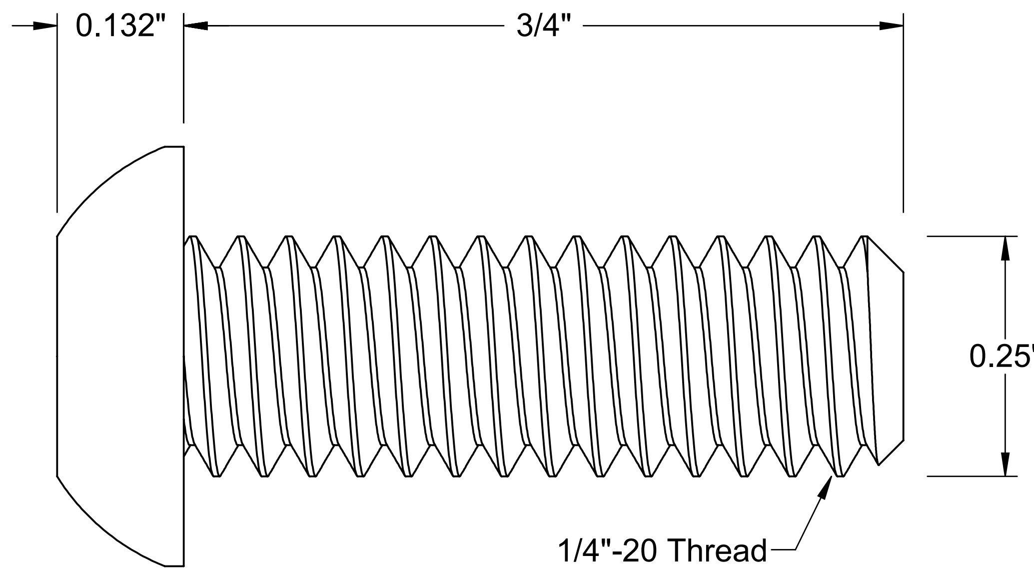
Alloy Steel Socket Head Cap Screw



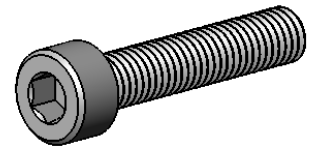
McMASTER-CARR CAD	PART NUMBER 91255A537
http://www.mcmaster.com © 2019 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Alloy Steel Button Head Hex Drive Screw

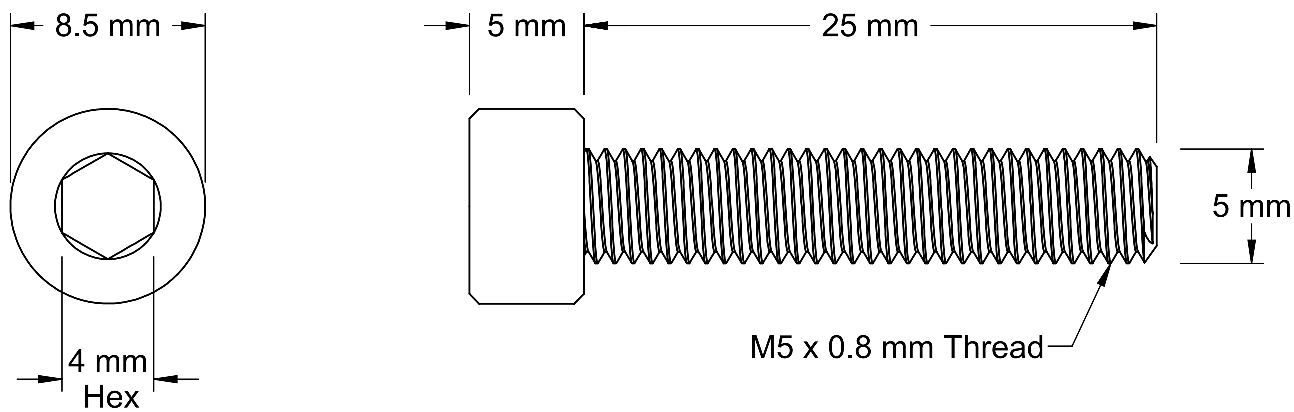






MCMASTER-CARR CAD	PART NUMBER 91255A540
http://www.mcmaster.com © 2019 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Alloy Steel Button Head Hex Drive Screw



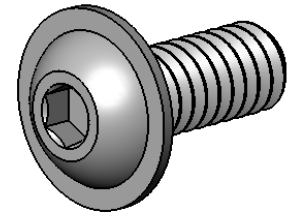


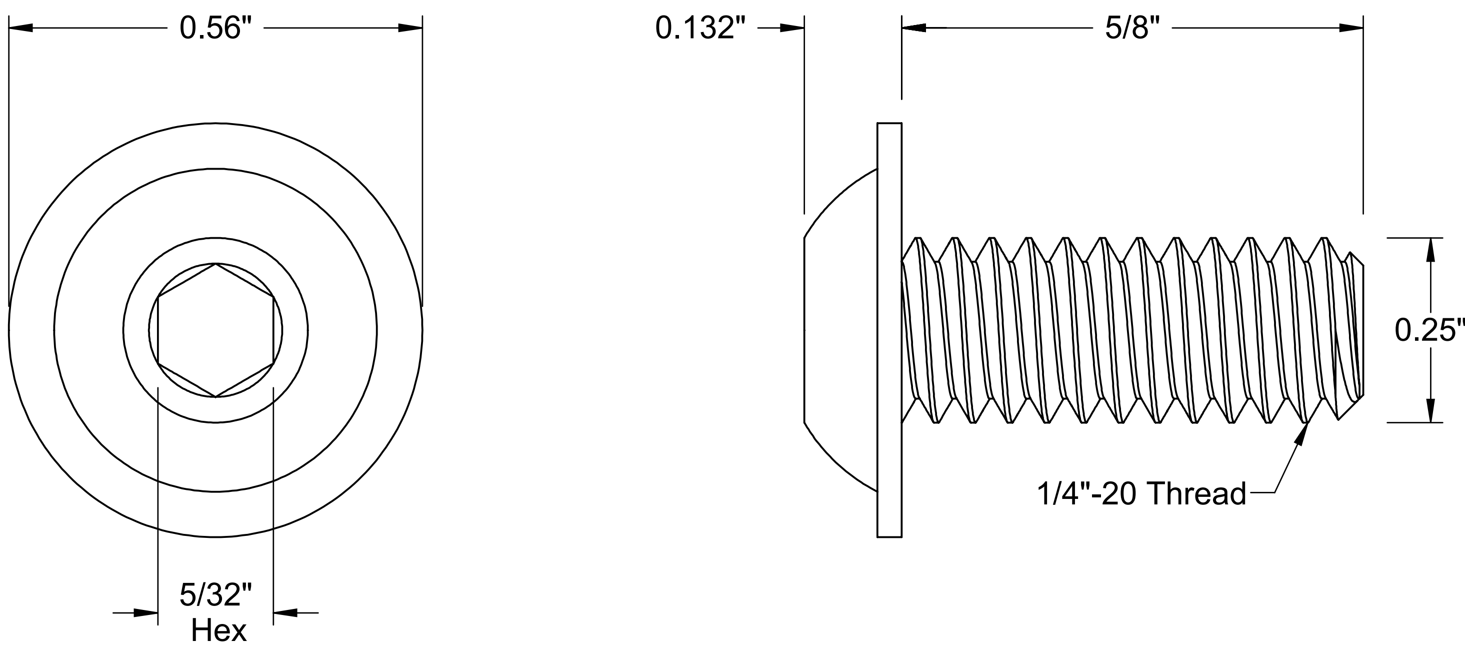
McMASTER-CARR® CAD

PART NUMBER **91292A129**

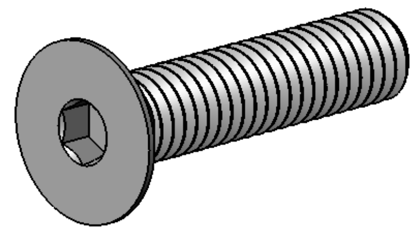
http://www.mcmaster.com

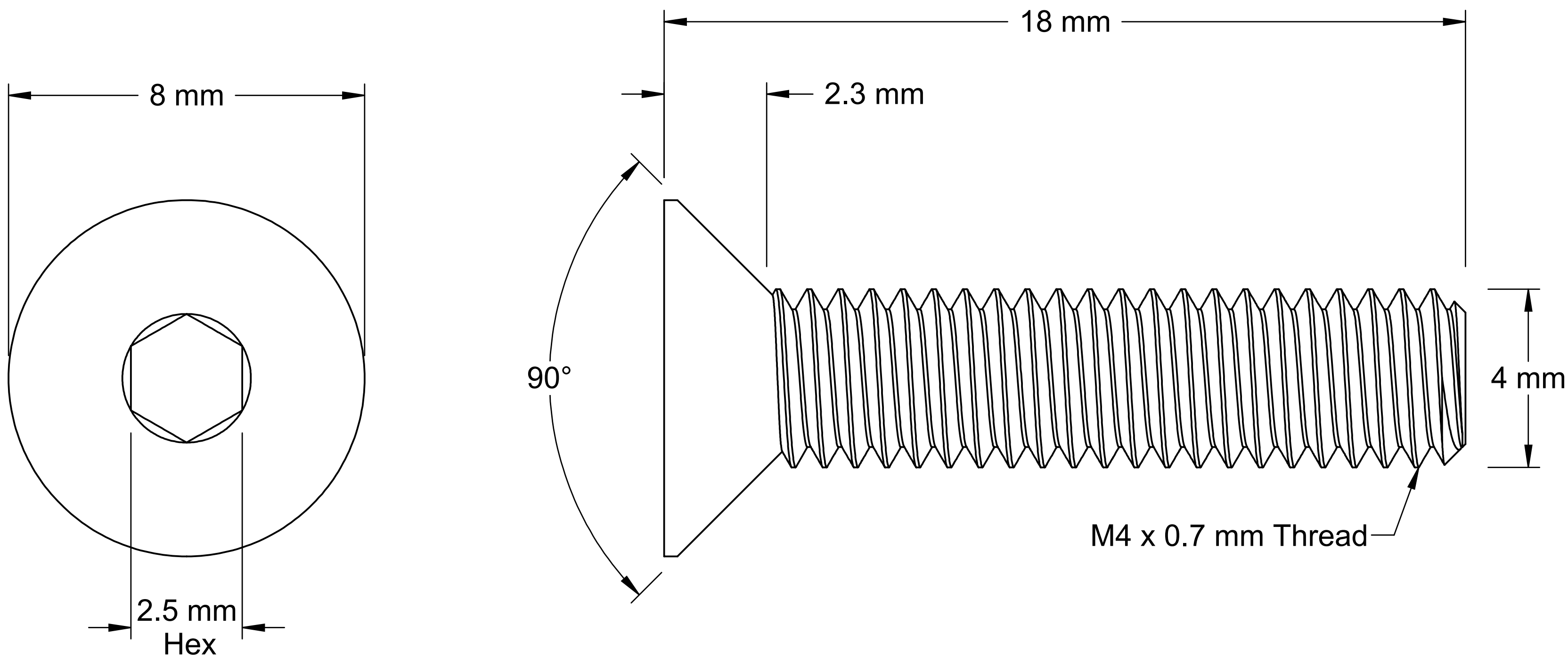
Stainless Steel
Socket Head Cap Screw



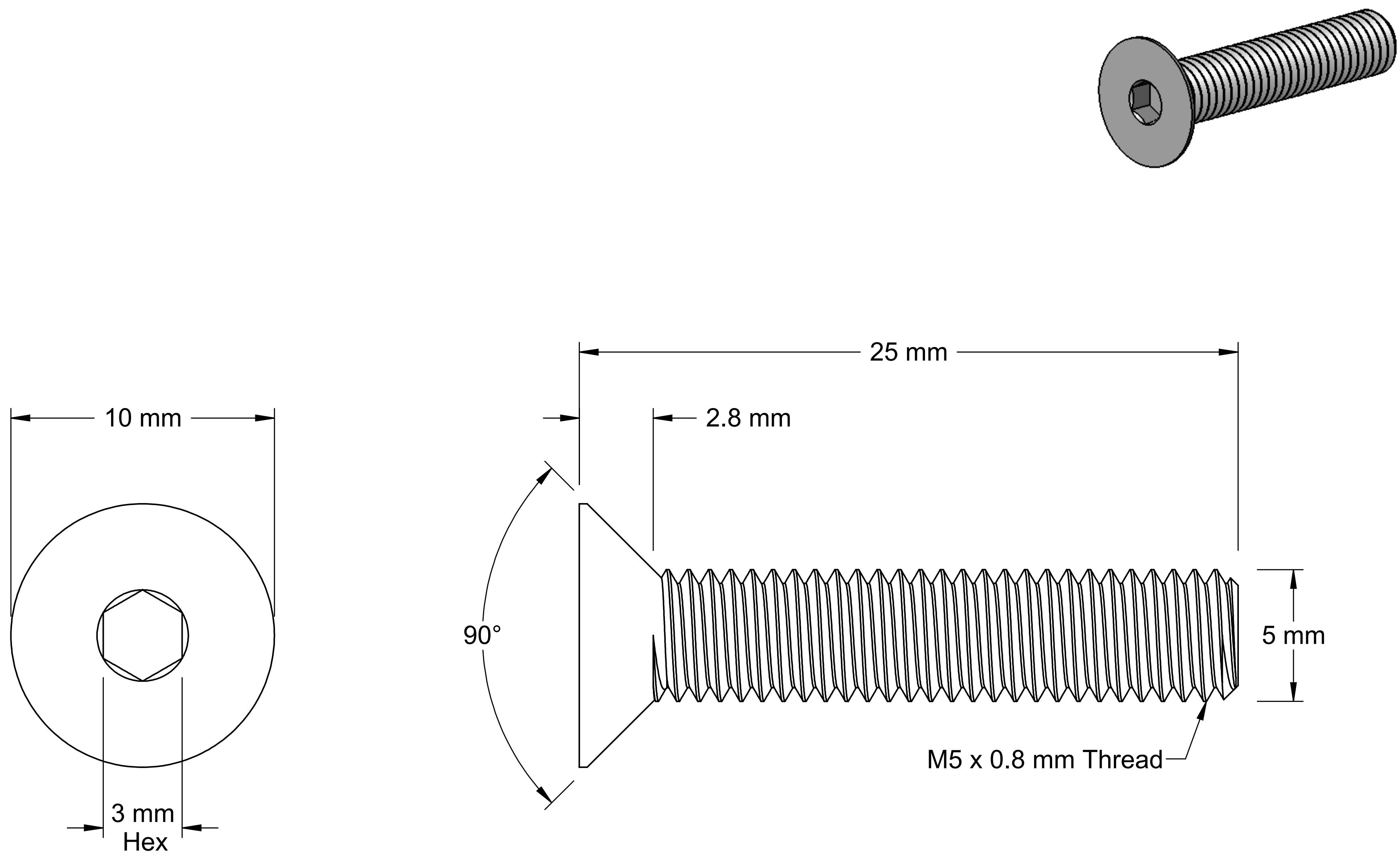


MCMASTER-CARR® CAD	PART NUMBER 91355A178
http://www.mcmaster.com © 2014 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Flanged Alloy Steel Button-Head Socket Cap Screw

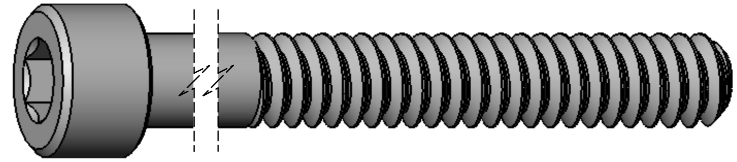


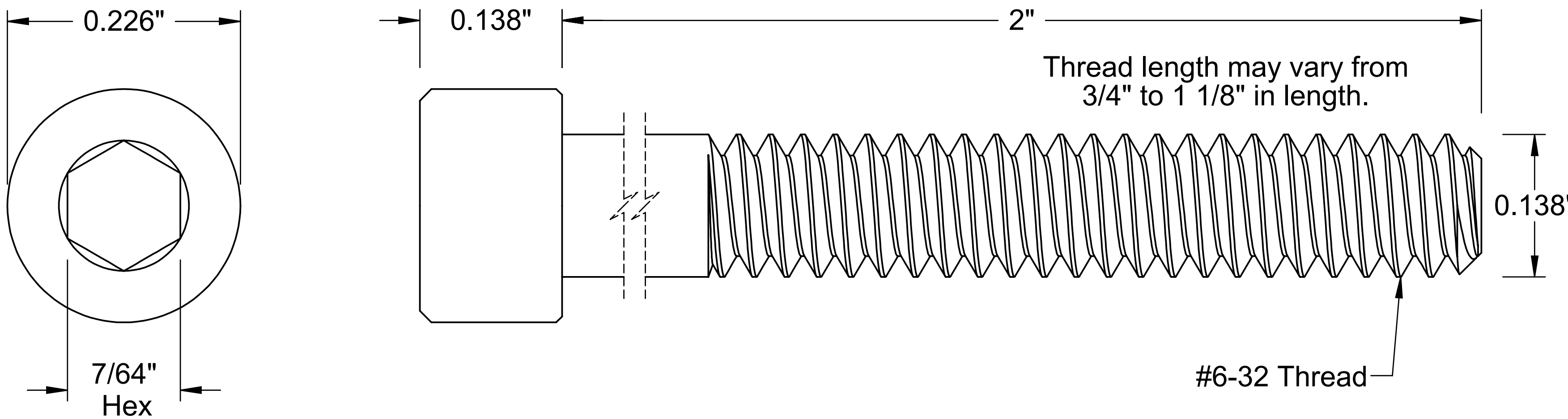


McMASTER-CARR® CAD	PART NUMBER 92125A196
http://www.mcmaster.com	Stainless Steel Flat-Head Socket Cap Screw

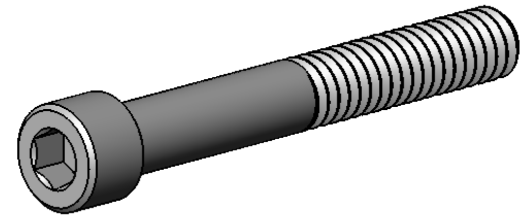


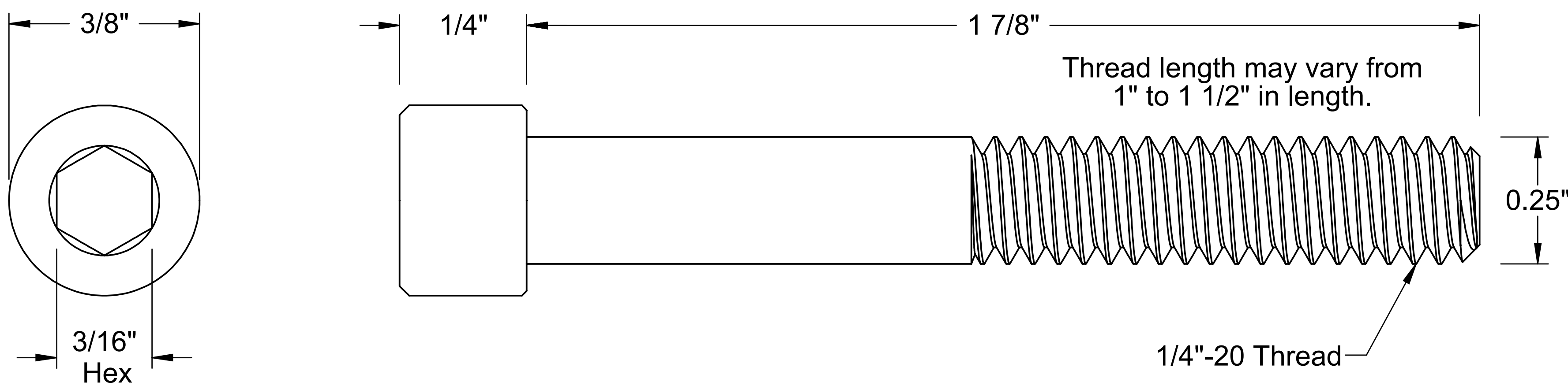
McMASTER-CARR®	PART NUMBER	92125A216
http://www.mcmaster.com © 2014 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Stainless Steel Flat-Head Socket Cap Screw	





McMASTER-CARR CAD	PART NUMBER 92196A159
http://www.mcmaster.com © 2014 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Stainless Steel Socket Head Cap Screw

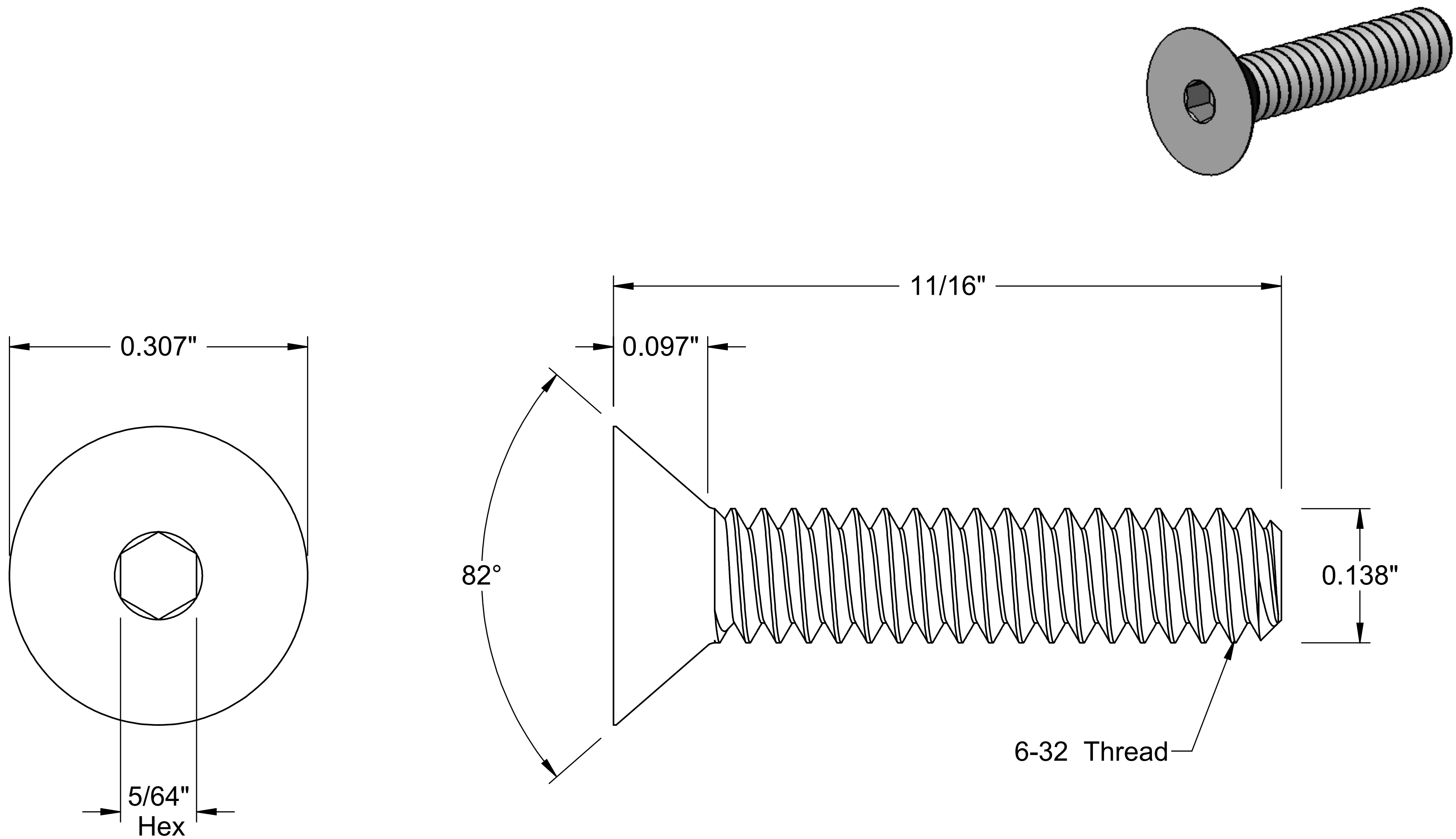




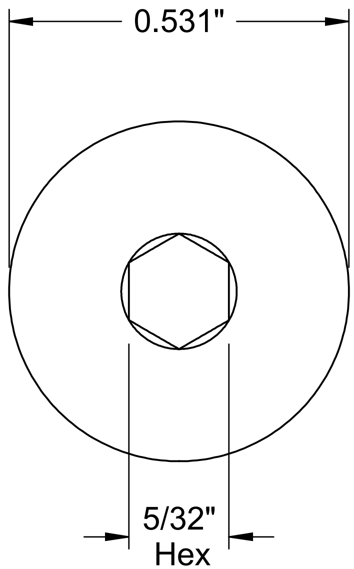
McMASTER-CARR® CAD

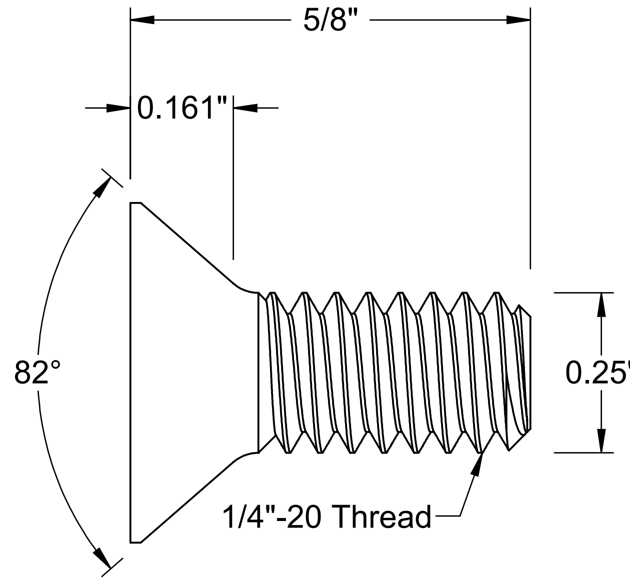
PART NUMBER 92196A305

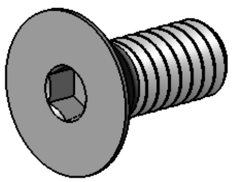
Stainless Steel
Socket Head Cap Screw



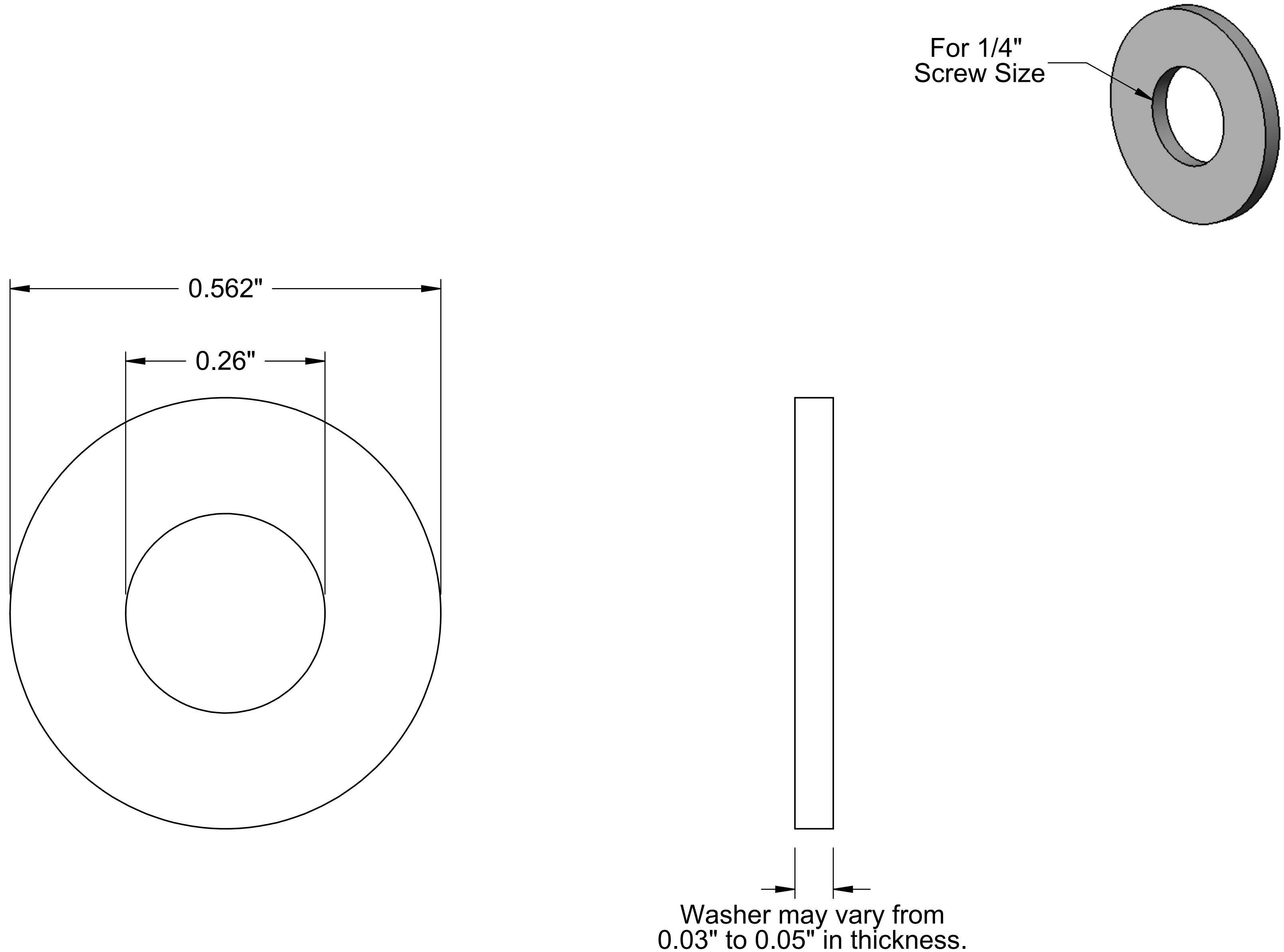
McMASTER-CARR® CAD	PART NUMBER	92210A164
http://www.mcmaster.com © 2019 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	18-8 Stainless Steel Hex Drive Flat Head Screw	







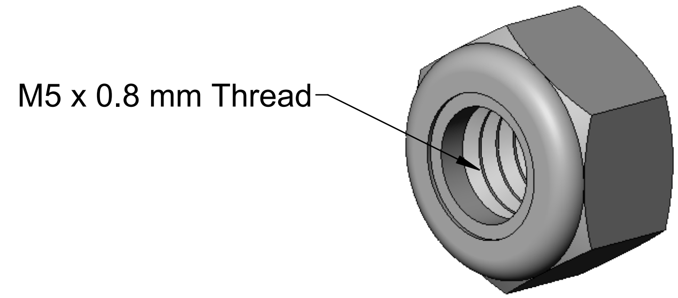
McMASTER-CARR® CAD	PART NUMBER 92210A539
http://www.mcmaster.com © 2014 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Stainless Steel Flat-Head Socket Cap Screw

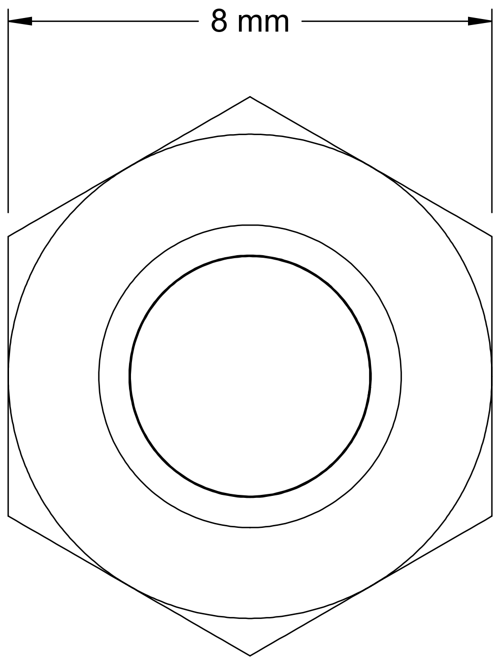


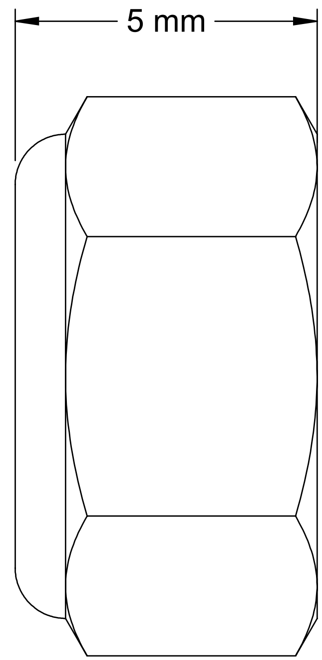
McMASTER-CARR® CAD

PART NUMBER **92917A155**

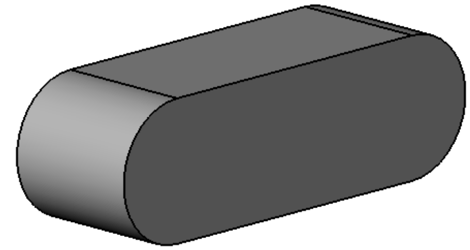
General Purpose Washer

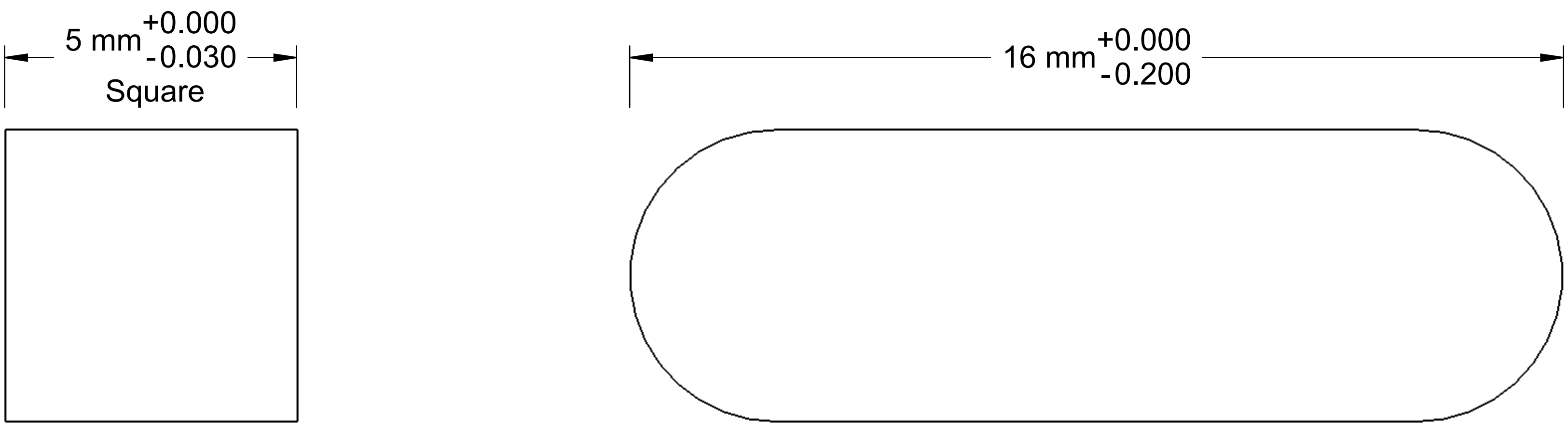






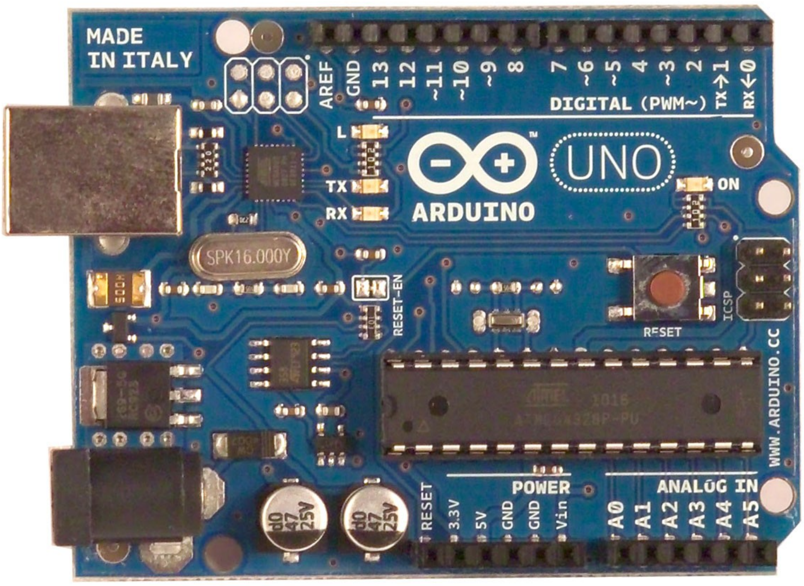
McMASTER-CARR® CAD	PART NUMBER 93625A200
http://www.mcmaster.com © 2020 McMaster-Carr Supply Company	Metric Nylon-Insert Locknut
Information in this drawing is provided for reference only.	





McMASTER-CARR® CAD	PART NUMBER	96717A160
http://www.mcmaster.com	Rounded Machine Key	

Arduino UNO



Product Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduno, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Index

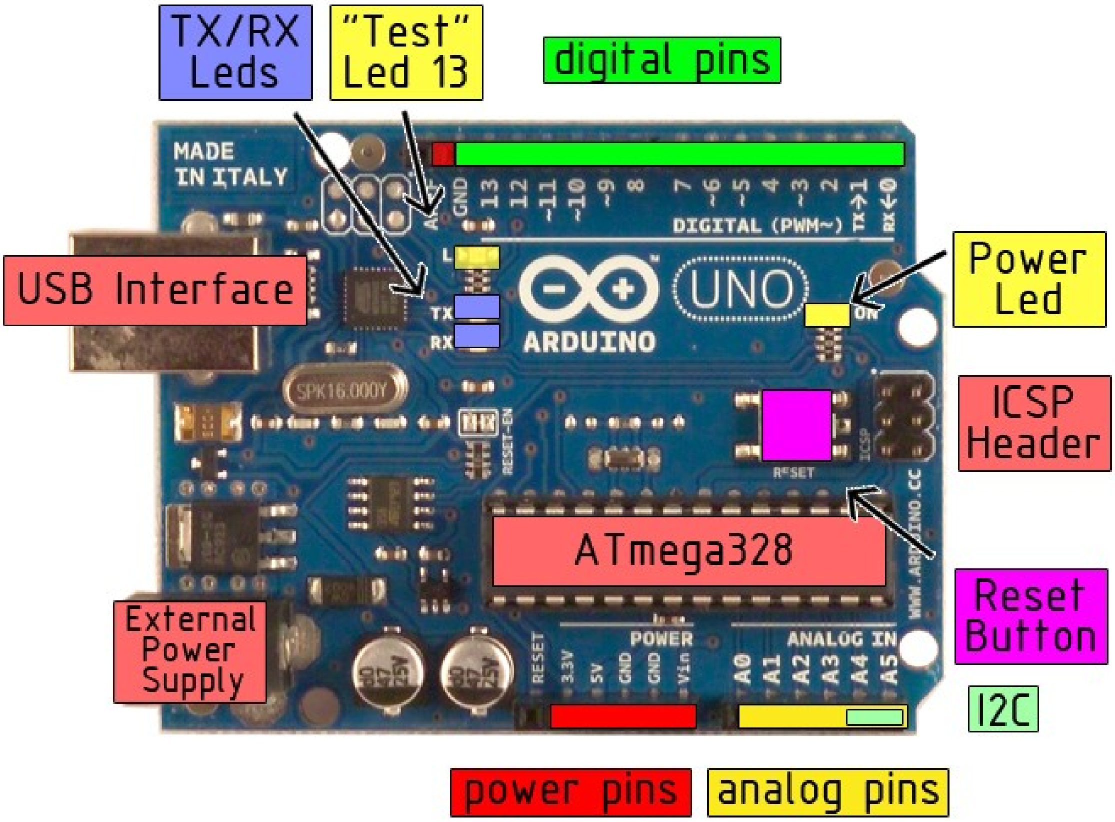
Technical Specification

EAGLE files: arduino-duemilanove-uno-design.zip Schematic: arduino-uno-schematic.pdf

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB of which 0.5 KB used by bootloader
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

the board



Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. TThese pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **I^2C: 4 (SDA) and 5 (SCL).** Support I^2C (TWI) communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and Atmega328 ports.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required..

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno w/ ATmega328" from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega8U2 firmware source code is available . The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader).

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

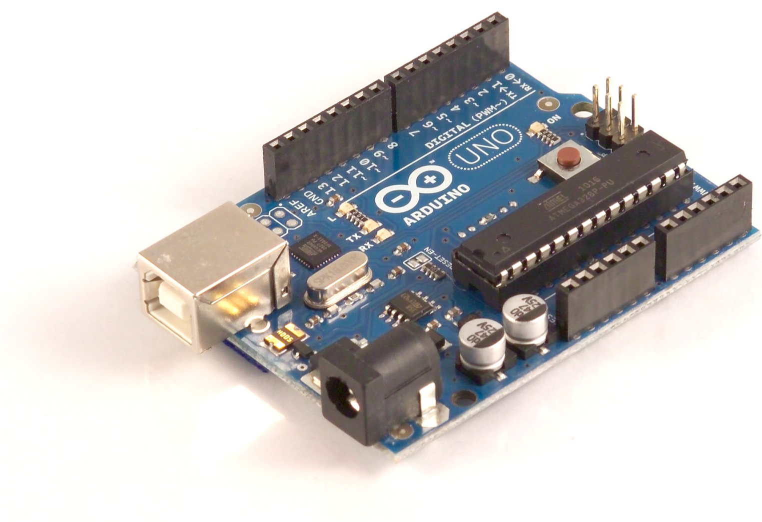
The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.



How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Linux Install

Windows Install

Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

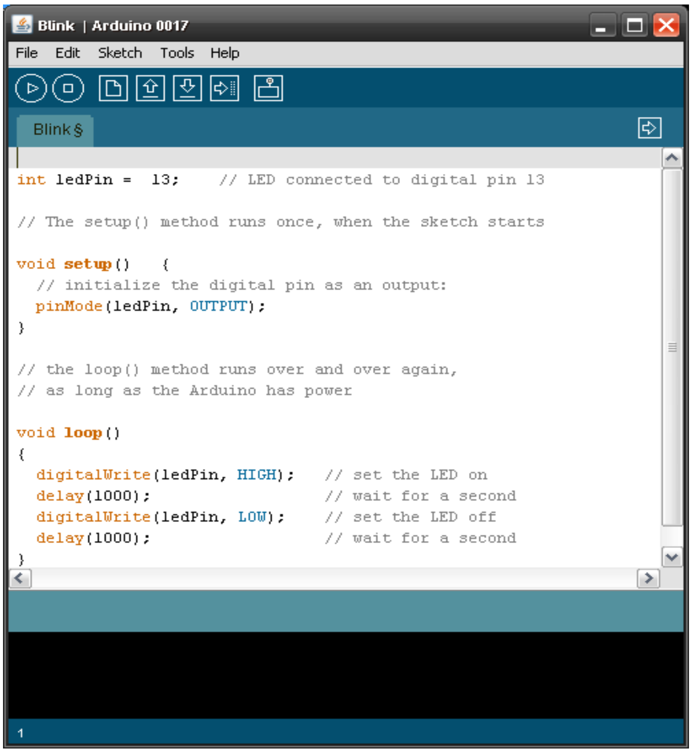
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

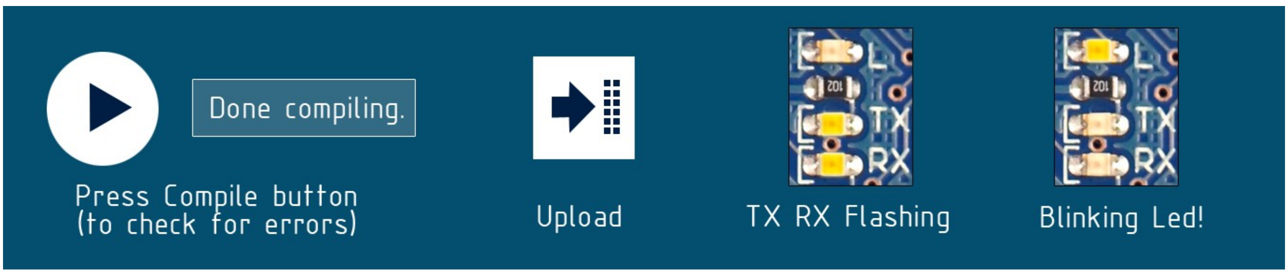
**File>Sketchbook>
Arduino-0017>Examples>
Digital>Blink**

Once you have your skecth you'll see something very close to the screenshot on the right.

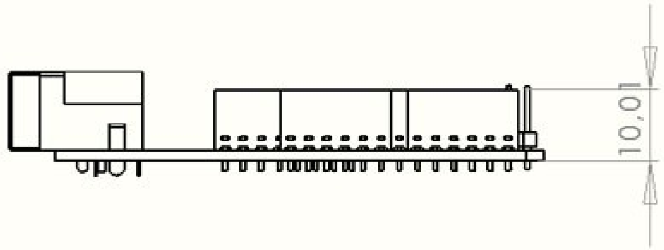
In **Tools>Board** select

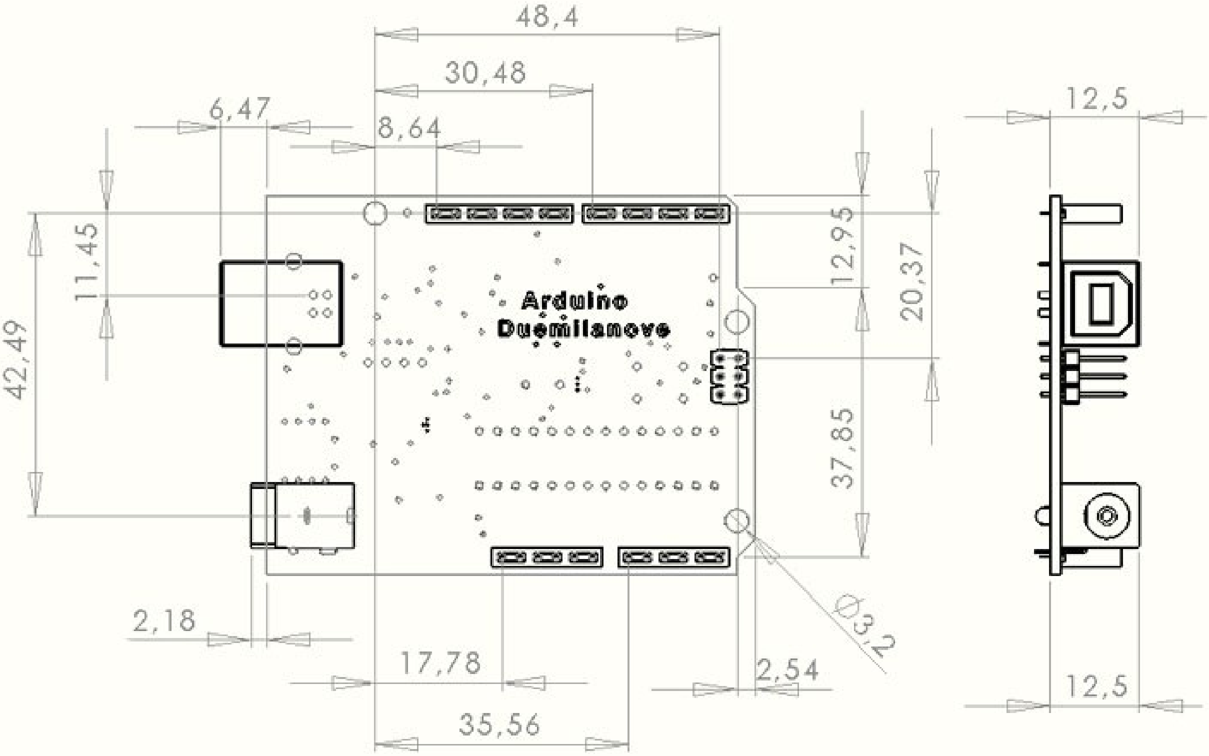
Now you have to go to
Tools>SerialPort
and select the right serial port, the one arduino is attached to.

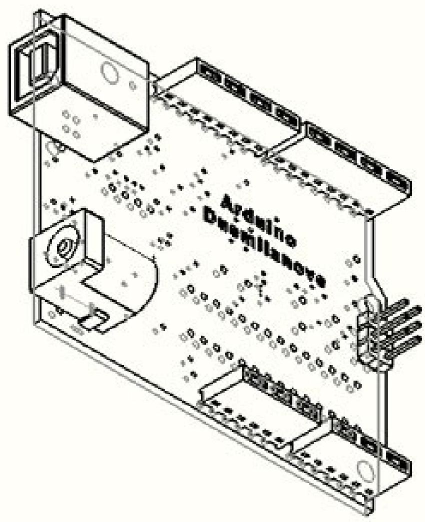




Dimensioned Drawing







Terms & Conditions

ARDUINO

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

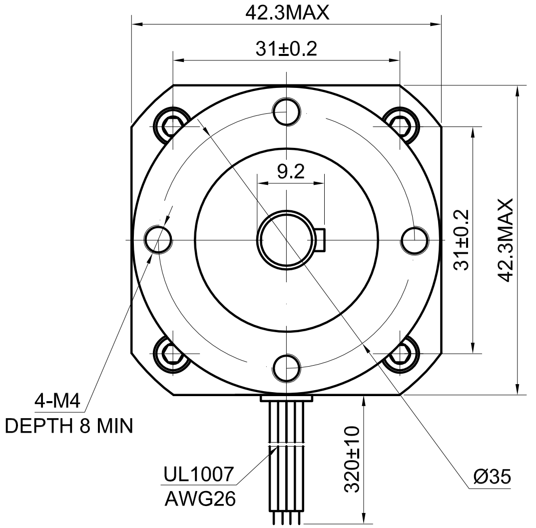
4. Changes to specifications

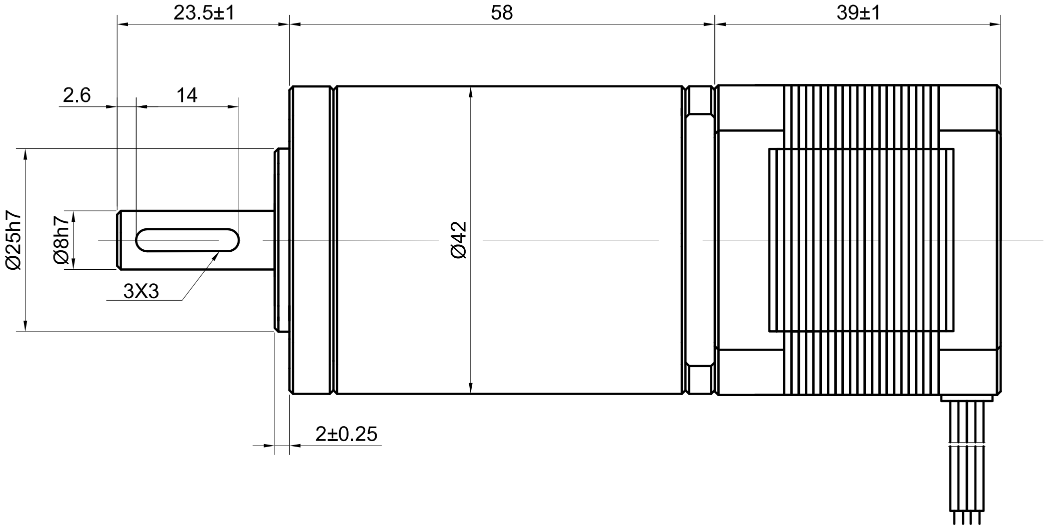
The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

IMPATTO ZERO® www.lifegate.it LIFEGATE

The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest's.



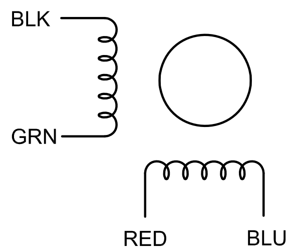


SPECIFICATION / CONNECTION	BIPOLAR
AMPS/PHASE	1.68
RESISTANCE/PHASE(Ohms)@25°C	1.60±10%
INDUCTANCE/PHASE(mH)@1KHz	3.20±20%
HOLDING TORQUE w/o GEARBOX(Nm)[lb-in]	0.39[3.45]
GEAR RATIO	100
EFFICIENCY	90.00%
STEP ANGLE w/o GEARBOX(°)	1.80
BACKLASH@NO-LOAD(arcmin)	<=25
MAX.PERMISSIBLE TORQUE(Nm)	15.00
MOMENT PERMISSIBLE TORQUE(Nm)	25.00
WEIGHT(kg)	0.88
LIFE(h)	6000.00
AMBIENT TEMPERATURE(°C)	-10 ~ 50
INSULATION CLASS	B
NOISE(dB)	<=50
MAX. RADIAL LOAD(N)	150

TYPE OF CONNECTION (EXTERN)		MOTOR	
PIN NO	BIPOLAR	LEADS	WINDING
1	A —	BLK	A
2	A\ —	GRN	A\
3	B —	RED	B
4	B\ —	BLU	B\

FULL STEP 2 PHASE-Ex. ,
WHEN FACING MOUNTING END (X)

STEP	A	B	A\	B\	CW	CCW
1	+	+	-	-	↓	↑
2	-	+	+	-		
3	-	-	+	+		
4	+	-	-	+		

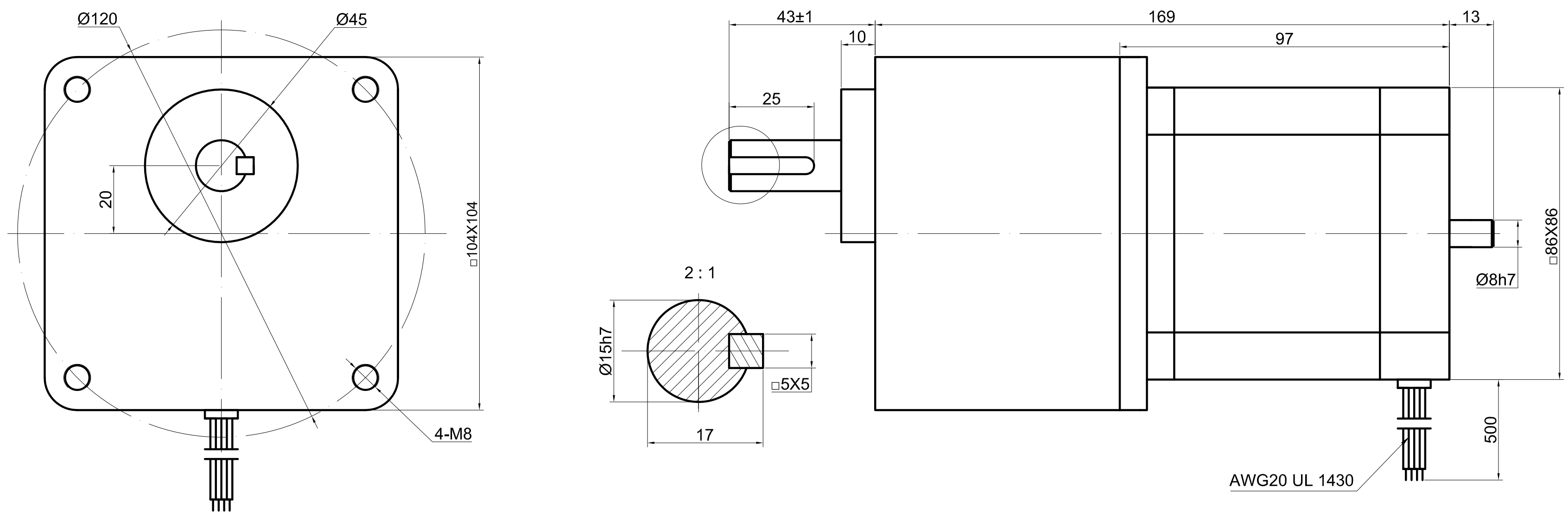


STEPPERONLINE®

	APVD		8.18.2018
	CHKD		
1:1	DRN		
SCALE	SIGNATURE		DATE

STEPPER MOTOR

17HS15-1684S-HG100

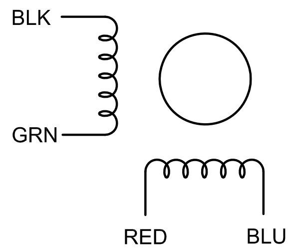


SPECIFICATION \ CONNECTION	BIPOLAR
AMPS/PHASE	4.00
RESISTANCE/PHASE(Ohms)@25°C	0.60±10%
INDUCTANCE/PHASE(mH)@1KHz	4.50±20%
HOLDING TORQUE w/o GEARBOX(Nm)[lb-in]	4.50[39.83]
GEAR RATIO	20.00
EFFICIENCY	70.00%
STEP ANGLE w/o GEARBOX(°)	1.80
BACKLASH@NO-LOAD	<=3°
MAX.PERMISSIBLE TORQUE(Nm)	40.00
WEIGHT(Kg)[lb]	5.00[11.02]
ROTOR INERTIA w/o GEARBOX(g-cm²)	1800.00
INSULATION RESISTANCE 100 Mohm(UNDER NORMAL TEMPERATURE AND HUMIDITY)	
DIELECTRIC STRENGTH 500VAC FOR 1MIN.(BETWEEN THE MOTOR COILS AND THE MOTOR CASE)	
TEMPERATURE RISE:MAX.80°C（MOTOR STANDSTILL;FOR 2PHASE ENERGIZED）	
AMBIENT TEMPERATURE -10°C~50°C[14°F~122°F]	
INSULATION CLASS B 130°C[266°F]	

TYPE OF CONNECTION (EXTERN)		MOTOR	
PIN NO	BIPOLAR	LEADS	WINDING
1	A —	BLK	A
2	A\ —	GRN	A\
3	B —	RED	B
4	B\ —	BLU	B\

FULL STEP 2 PHASE-Ex. ,
WHEN FACING MOUNTING END (X)

STEP	A	B	A\	B\	CW	CCW
1	+	+	-	-	↓	↑
2	-	+	+	-		
3	-	-	+	+		
4	+	-	-	+		



STEPPERONLINE®		APVD		8.20.2018
		CHKD		
	1:2	DRN		
	SCALE	SIGNATURE		DATE

STEPPER MOTOR

34HS38-4004D-SG20

Power Supply Adapter

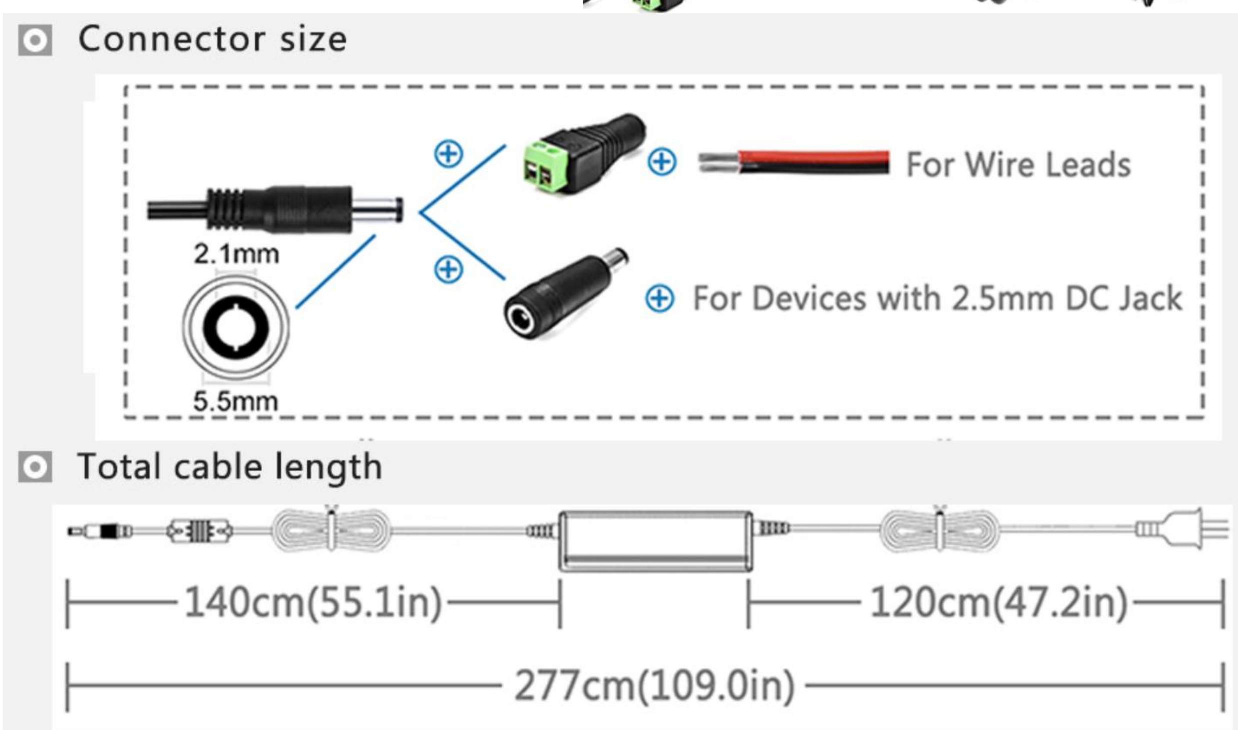
AC/DC Power Transformer

Specifications

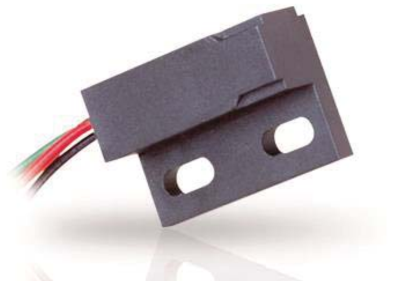
24V

10A

240W



Magnetic Proximity Sensors (Reed)



MP2019 Sensors

Reed based magnetic proximity sensor
in plastic flange-mount package

Description

The MP2 Series sensors are reed position sensors. Normally open or closed contacts change states when a magnetic field is applied. The sensors act as non-latching electrical switches.

Features

- Contacts hermetically sealed for long life
- Zero power consumption
- Suitable for DC and AC circuits
- Flange mount
- RoHS compliant
- IP65
- Operate/Release Distances*: 10.16 mm – 22.86 mm (0.4” – 0.9”)

Typical Applications

- Door position & interlock
- Limit switch
- Flow/speed
- Home security
- Pedal switch

Environmental Specifications

Vibration	20 g 10 Hz to 1 kHz (MP201901), 20 g 10 Hz to 55 Hz (MP201902, M201903)
Operating Temperature	-40 °C to 105 °C (-40 °F to 221 °F)
Storage Temperature	-40 °C to 105 °C (-40 °F to 221 °F)
Ingress Protection	IP65

Electrical Specifications

Typical Operating Time	1 ms
Breakdown Voltage	200 VDC min.
Switching Voltage and Current	See Products chart
Contact Resistance and Power rating:	See Products chart

Mechanical Specifications

Housing Material	Glass-reinforced plastic
Operate/Release Distances* *with AS201901 magnetic actuator	10.16 mm – 22.86 mm (0.4” – 0.9”)

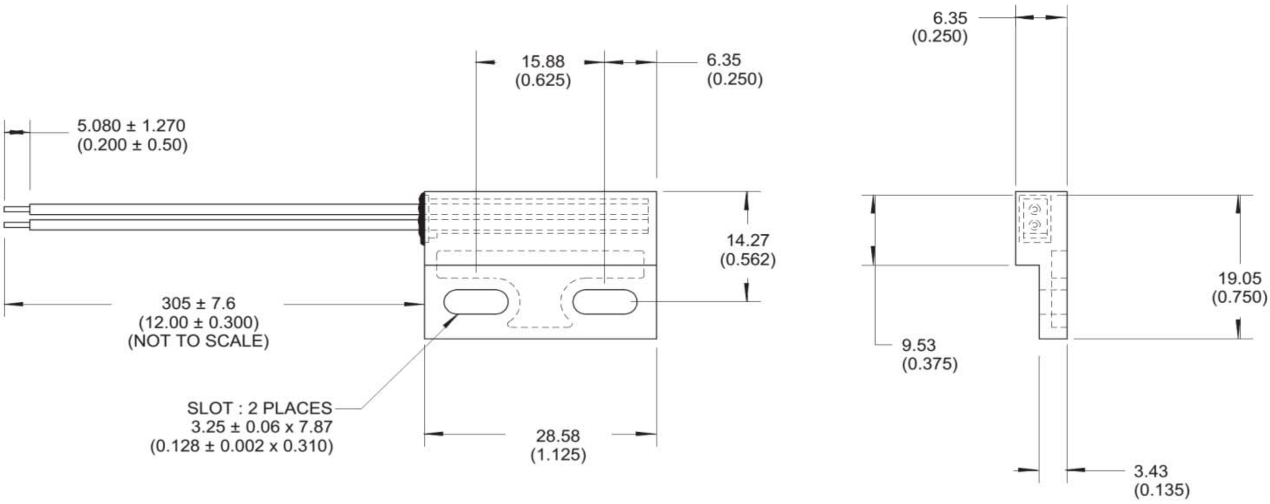
Products

Part Number	Contact Form	Maximum Power Rating	Maximum Switching Voltage	Maximum Switching Current	Maximum Contact Resistance
MP201901	SPST-NO Form A	10 W	175 VAC/VDC	0.5 A	0.200 Ω
MP201902	SPST-NC Form B	3 W	30 VAC/VDC	0.2 A	0.100 Ω
MP201903	SPDT-CO Form C	3 W	30 VAC/VDC	0.2 A	0.100 Ω

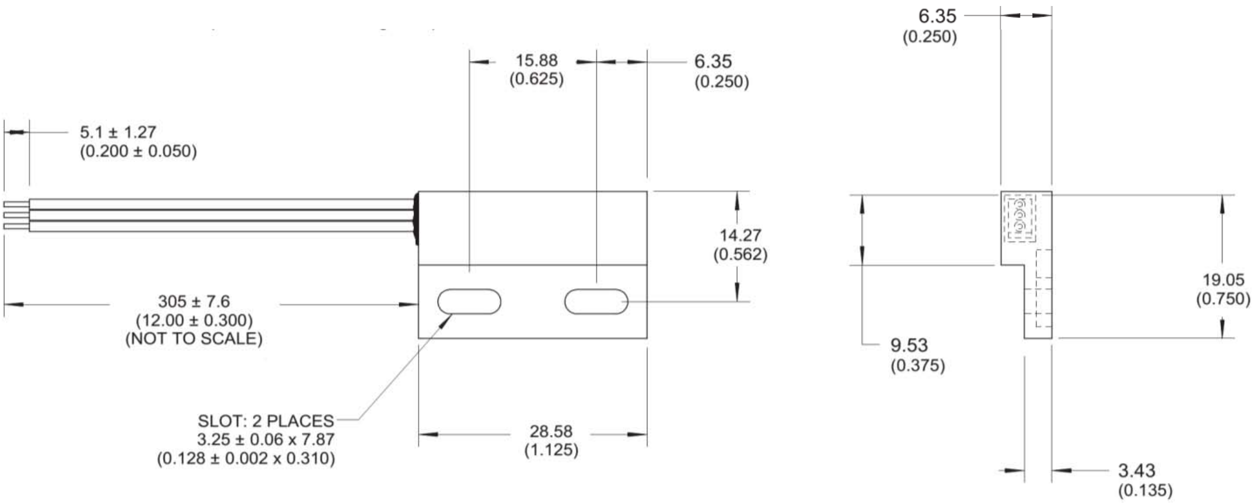
All MP2019 sensors come with leads 24 AWG x 305 mm (12”)

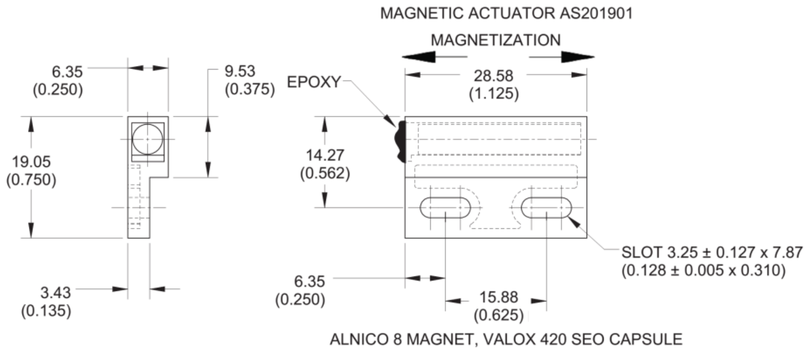
Dimensions mm (inches)

MP201901 (Form A - N.O.) and MP201902 (Form B - N.C.)



MP201903 (Form C - SPDT Changeover)

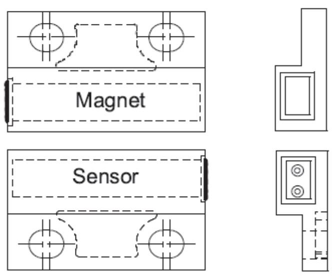




Wire designations

BLACK	NO
BLUE	NC
BROWN	COMMON

Alignment for Best Performance



CNBTR 6-Wire Slip Ring

Specifications:

Material: Metal and plastic

Color: Blue & Black

Pathway: 6 Wires

Brand name:CNBTR

Cable length: 200MM

Hole Dia: 12.7MM

Body Dia: 54MM

Rated voltage:380V

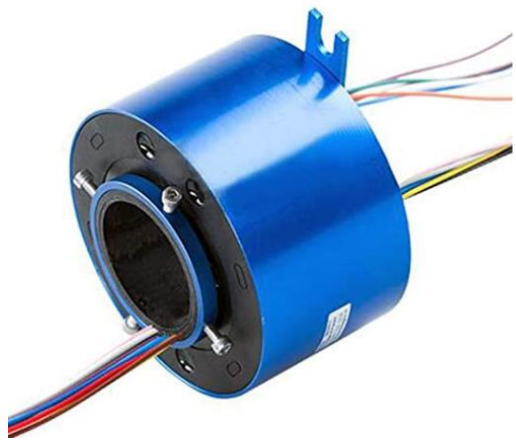
Rated Current:10A per circuit

Size: 36 x 54MM/1.42 x 2.13 Inch (L x Dia)

Weight:127g

Package Include: 1 xThrough Hole Slip Ring

Taidacent Slip Ring



Product description

Style:**12 Wire 10A Inner hole 25.4mm OD 86mm**

Product Name: Through Hole Slip Ring
Speed: 250 rev / min
Protection level: IP54
Insulation resistance: 500MQ@500VDC
Working life: ≥ 5 million rpm
Bearing: high precision ball bearing
Wire Specifications: Please consult us
Mechanical vibration: MIL-SID-810E
Number of channels: 1-36 road
Rated current: 10A
Rated voltage: 380V
Contact material: precious metal
Housing Material: Engineering Plastics
Torque: 0.01N.M
Working temperature: -40 ° C ~ +80 ° C
Relative humidity: 60%
Wire length: 300mm
Product features
Gold-gold contact points ensure stability and service life, mainly for civil and commercial applications
Smooth operation, compact structure, low torque, compatible with data bus protocol
The use of colored conductors for stator and rotor lines makes circuit connections simpler
The conductive ring is designed with a 90 degree angle V row groove for smooth rotation, low torque and low electrical noise.
Product monolithic structure, advanced fiber brush technology, branch temple power, signal mixed transmission
Life depends on the working environment and speed
Application field
cloud operation
Electrical test equipment
Manufacturing and processing control equipment
Medical equipment
CCTV Security Monitoring Equipment
Robot
Exhibition / Display Equipment
Rotating table
Small wind turbines
Radar antenna

STEPPERONLINE®

User Manual for
Full Digital Stepper Driver

Y Series Open-loop
Stepper Driver

Version 1.0
STEPPERONLINE®

Attention: Please read this manual carefully before using the driver.

1 Introductions & Features

The Y series Open-loop drivers are divided according to the driving power, which are DM420Y, DM542Y, DM556Y, DM860Y. The Y series open loop drivers are digital stepper drivers with simple design and easy setup. By implementing the advanced stepper control technology, this stepper drive is able to power 2-phase and 4 phase stepper motors smoothly with optimal torque and low motor heating & noise. All the micro step and output current are done via DIP switches. Therefore, the Y series open loop drivers are ideal choices for applications requiring simple step & direction control of NEMA 8, 11, 14, 16, 17, 23, 24 and 34 stepper motors. And features are showing below:

- Anti-Resonance for optimal torque, extra smooth motion, low motor heating and noise
- Motor auto-identification and parameter auto-configuration for optimal torque from wide-range motors
- Step & direction (PUL/DIR) control
- Multi-Stepping for smooth motor movement
- TTL compatible and optically isolated inputs
- Input voltage DC18~36V, DC20~50V, DC36~110V(AC18~80V)
- Micro-step resolutions: 200~6400, 200~40000, 400~40000
- Output current settings: 0.3~2.2A, 1~4.2A, 1.7~5.6A, 2.4~7.2A
- Soft-start with no “jump” when powered on
- Automatic idle-current reduction
- Protections for over-voltage and over-current

2 Specifications

2.1 Operating Environment and other Specifications (Tj = 25°C/77°F)

Cooling	Natural Cooling or Forced Cooling	
Operating Environment	Environment	Avoid dust, oil fog and corrosive gases
	Ambient Temperature	0°C ~ 65°C (32°F ~ 149°F)
	Humidity	40%RH ~ 90%RH
	Operating Temperature	0°C ~ 50°C (32°F ~ 122°F)
	Vibration	10-50Hz / 0.15mm
Storage Temperature	-20°C ~ 65°C (-4°F ~ 149°F)	
Weight	120g(DM420Y), 300g(DM542Y), 300g(DM556Y), 500g(DM860Y)	

2.2 Electrical Specifications and Mechanical Specifications of different models

■ DM420Y

Electrical Specifications

Parameters	DM420Y			
	Min	Typical	Max	Unit
Output Peak Current	0.3(0.2 RMS)	-	2.2(1.6 RMS)	A
Supply Voltage	18	24	36	VDC
Logic Signal Current	7	10	15	mA
Pulse input frequency	0	-	100	kHz
Minimal pulse width	5	-	-	µS
Minimal direction setup	5	-	-	µS
Isolation resistance	500			MΩ

Mechanical Specifications (unit: mm)

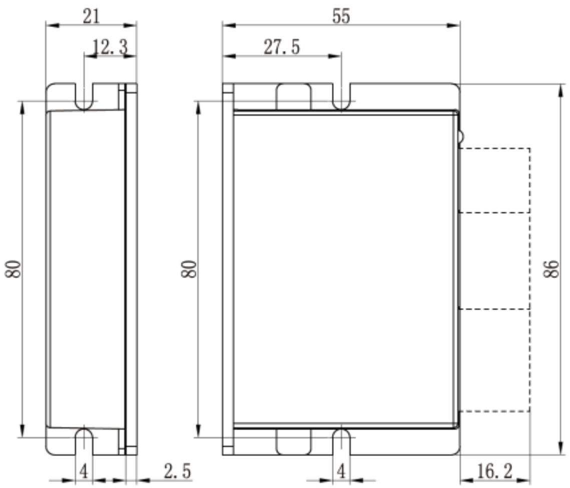


Figure 1: Mechanical specifications

Recommend use side mounting for better heat dissipation

STEPPERONLINE

■ DM542Y

Electrical Specifications

Parameters	DM542Y			
	Min	Typical	Max	Unit
Output Peak Current	1.0(0.71 RMS)	-	4.2(3.0 RMS)	A
Supply Voltage	20	36	50	VDC
Logic Signal Current	7	10	15	mA
Pulse input frequency	0	-	200	kHz
Minimal pulse width	2.5	-	-	µS
Minimal direction setup	2.5	-	-	µS
Isolation resistance	500			MΩ

Mechanical Specifications (unit: mm)

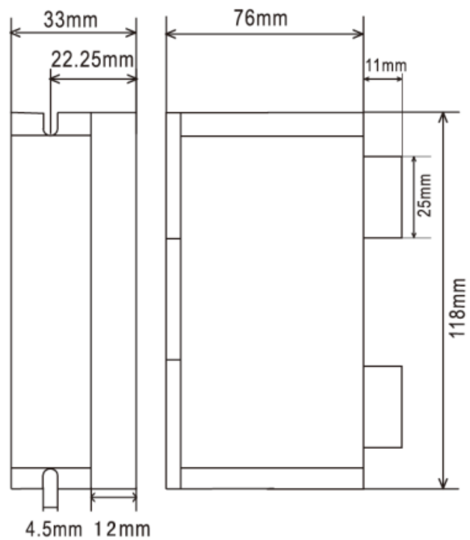


Figure 2: Mechanical specifications

Recommend use side mounting for better heat dissipation

■ DM556Y

Electrical Specifications

Parameters	DM556Y			
	Min	Typical	Max	Unit
Output Peak Current	1.7(1.2 RMS)	-	5.6(4.0 RMS)	A
Supply Voltage	20	36	50	VDC
Logic Signal Current	7	10	15	mA
Pulse input frequency	0	-	200	kHz
Minimal pulse width	2.5	-	-	µS
Minimal direction setup	2.5	-	-	µS
Isolation resistance	500			MΩ

Mechanical Specifications (unit: mm)

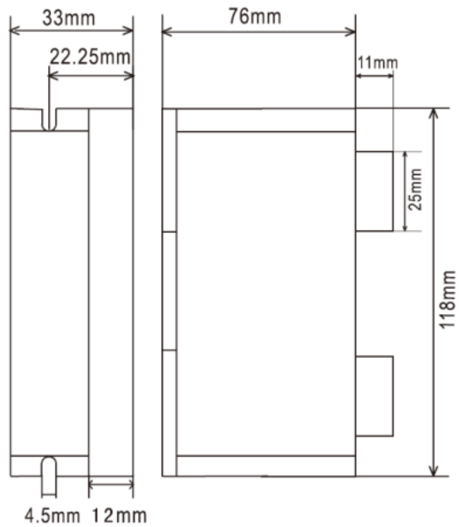


Figure 3: Mechanical specifications

Recommend use side mounting for better heat dissipation

■ DM860Y

Electrical Specifications

Parameters	DM860Y			
	Min	Typical	Max	Unit
Output Peak Current	2.4(2.0 RMS)	-	7.2(6.0 RMS)	A
Supply Voltage	18(24)	-	80(110)	VAC(VDC)
Logic Signal Current	7	10	15	mA
Pulse input frequency	0	-	200	kHz
Minimal pulse width	2.5	-	-	µS
Minimal direction setup	2.5	-	-	µS
Isolation resistance	500			MΩ

Mechanical Specifications (unit: mm)

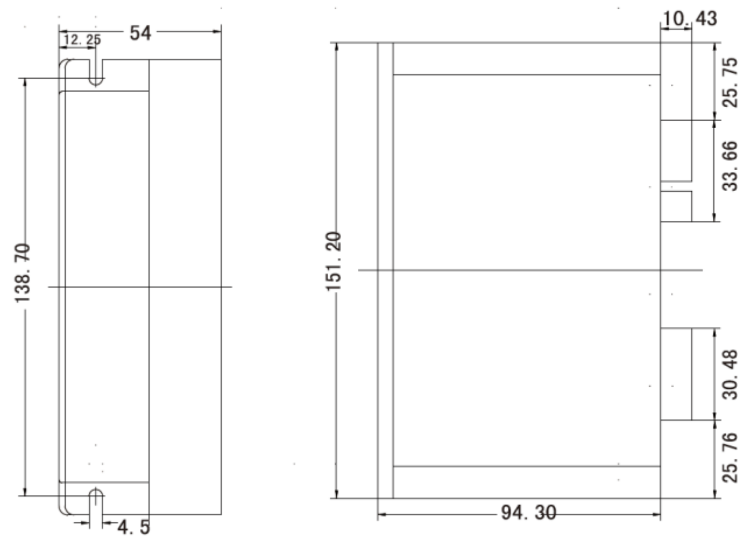


Figure4: Mechanical specifications

Recommend use side mounting for better heat dissipation

3 Pin Assignment and Description

3.1 Connector P1 Configurations

■ DM420Y:

Pin Function	Details
PU	The falling edge is valid and the pulse goes one step when the pulse goes from high to low. Built-in input resistor 384 ohms. Requirement: Low level 0~0.5V, high level is consistent with common anode end, pulse width greater than 2.5 microseconds.
DR	Used to change the direction of the motor. Built-in input resistor 384 ohms. Requirement: Low level 0~0.5V, high level is consistent with common anode end, pulse width greater than 2.5 microseconds.
MF	When it is active（low level）, the motor coil current is turned off, the driver stops working, and the motor is in a free state. Built-in input resistor 384 ohms. Requirement: Low level 0~0.5V, high level is consistent with common anode end, pulse width greater than 2.5 microseconds.
+(5V)	+3.3~24V can be driven, higher than +5V must add current limiting resistor, 3.3~5V does not need to connect current limiting resistor, 24V current limiting resistor is connected to 2000 ohm, 12V current limiting resistor is connected to 820 ohm

■ DM542Y\DM556Y\DM860Y:

Pin Function	Details
PU+	Connected to the signal power supply, +5~+24V can be driven, higher than 24V need to be connected to the current limiting resistor in PU-.
PU-	The falling edge is valid and the pulse goes one step when the pulse goes from high to low. Requirements: Low level: 0~0.5V, high level 5~24V, pulse width greater than 2.5 microseconds.
DR+	Connected to the signal power supply, +5~+24V can be driven, higher than 24V need to be connected to the current limiting resistor in DR-.
DR-	Used to change the direction of the motor. Requirements: Low level: 0~0.5V, high level 5~24V, pulse width greater than 2.5 microseconds.
MF+	Connected to the signal power supply, +5~+24V can be driven, higher than 24V need to be connected to the current limiting resistor in MF-.
MF-	When it is active（low level）, the motor coil current is turned off, the driver stops working, and the motor is in a free state.

4 Wiring

4.1 Control Signal Connector (P1) Interface

The Y series open-loop drivers can accept differential and single-ended inputs (including open-collector and PNP output). The Y series open-loop drivers 3 optically isolated logic inputs which are located on connector P1 to accept line drive control signals. These inputs are isolated to minimize or eliminate electrical noises coupled with the drive control signals. Recommend using line drive control signals to increase noise immunity for the drive in interference environments. In the following figures, connections to open-collector and PNP signals are illustrated.

+5V : R1=0, R2=0
+12V : R1=510Ω , R2=820Ω
+24V : R1=1.2KΩ , R2=1.8KΩ

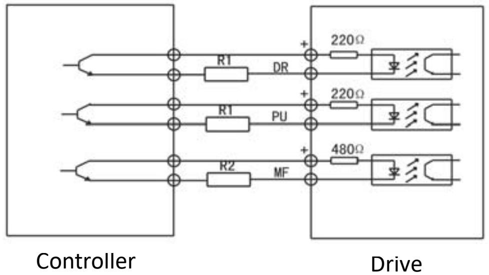


Figure 5: Differential wiring method

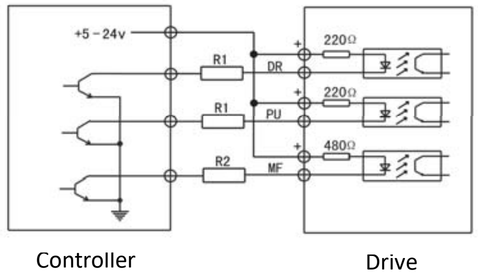


Figure 6: Independent common anode wiring method

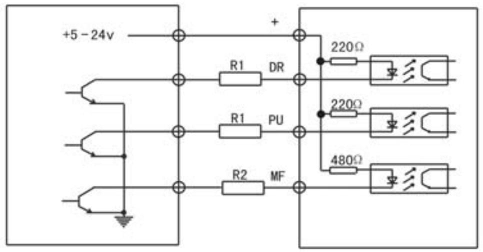


Figure 7: Common anode wiring method

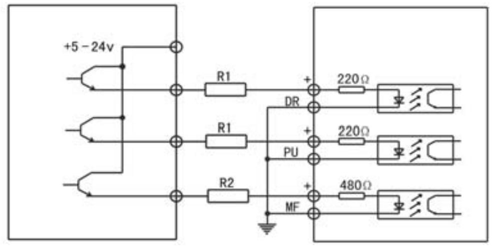


Figure 8: Independent common cathode wiring method

4.2 Connections of 4-lead Motor

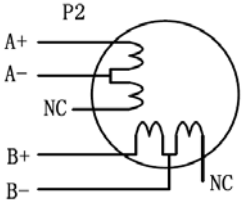


Figure 9: 4-lead Motor Connections

4.3 Connections of 6-lead Motor

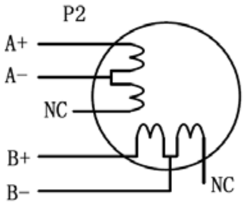


Figure10: 6-lead motor half coil (higher speed) connections

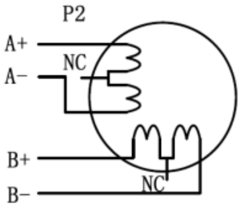


Figure 11: 6-lead motor full coil (higher torque) connections

4.4 Connections of 8-lead Motor

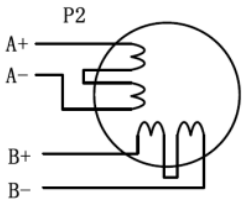


Figure 12: 8-lead motor series connections

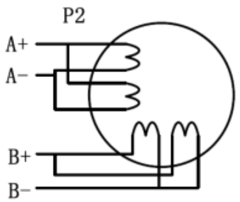


Figure 13: 8-lead motor parallel connections

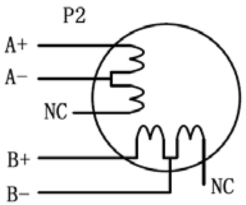


Figure14: 8-lead motor half coil

5 Frequently Asked Questions

Symptoms	Possible Problems
Motor is not rotating	No power
	Fault condition exists
	The driver is disabled
Motor rotates in the wrong direction	The direction signal level is reverse
Erratic motor motion	Control signal is too weak
	Control signal is interfered
	Wrong motor connection
	Something wrong with motor coil
Motor stalls during acceleration	Current setting is too small
	Motor is undersized for the application
	Acceleration is set too high
	Power supply voltage too low
Excessive motor and driver heating	Inadequate heat sinking / cooling
	Motor current setting is too high

6 Warranty

STEPPERONLINE® warrants its products against defects in materials and workmanship for a period of 12 months from shipment. During the warranty period, STEPPERONLINE will either, at its option, repair or replace products which proved to be defective. To obtain warranty service, a returned material authorization number (RMA) must be obtained before returning product for service.

Exclusions: The above warranty does not extend to any product damaged by reasons of improper or inadequate handlings by customer, improper or inadequate customer wirings, unauthorized modification or misuse, or operation beyond the electrical specifications of the product and/or operation beyond environmental specifications for the product.

For more detailed descriptions, please refer to the specifications on our website: www.omc-stepperonline.com.

OMC Corporation Limited
Address: #7 Zhongke Road, Jiangning
Nanjing, 211100 China
Tel: 0086-2587156578
Sales & Marketing: sales@stepperonline.com
Technical: technical@stepperonline.com
Web: www.omc-stepperonline.com

Appendix P: Design Verification Plan and Report (DVP&R)

Senior Project DVP&R													
Date:2/2/2020		Team: Automated Drone Calibration	Sponsor: Inspired Flight			Description of System: Gyroscope structure to allow for 3 axis rotation and drone orientation				DVP&R Engineer: Zach Richter			
TEST PLAN										TEST REPORT			
Item No	Specification #	Test Description	Acceptance Criteria	Test Responsibility	Test Stage	SAMPLES		TIMING		TEST RESULTS			NOTES
						Quantity	Type	Start date	Finish date	Test Result	Quantity Pass	Quantity Fail	
1	1	Compare setpoint angle to actual angle of device, measured by precallibrated IMU	less than 0.1 degree error	Ryan	FP	50	Sys	3/9/2020	4/15/2020				
2	7	Test the number of calibrations that end in errrors	less than 1% fail	Zach	FP	200	Sys	5/4/2020	5/20/2020				

Appendix Q. Test Procedures

Table 1. Engineering Specification Table

x	Specification Description	Requirement or Target (units)	Tolerance	Risk	Compliance
1	Accurate drone calibration	±1 degrees	Max	H	T
2	Number of set up steps	5	Max	M	I
3	Time to run	10 min	Max	M	I
4	Compatible drones	IF current and future drones	Y/N	M	I
5	Budget	$4,000	Max.	M	I
6	Weatherproof	Can be used outside	Y/N	M	I
7	Reliability	99%	Min	M	T
8	Size	4’ x 4’ base	Max	L	I
9	Lifetime	1,000 uses	Min	L	A
10	Weight	50 lbs.	Max	L	I
11	Fits through door	32” door frame	Y/N	L	I
12	Standard parts	When possible	Y/N	L	I
13	Uses wall power	120 V	+/- 5V	L	I

PASS FAIL

Test #1: Angular Accuracy of Device

Description of Test:

The purpose of this test is to ensure that the step inputs are providing the desired angular position for the device, since the motors run in open loop, meaning there is no positional feedback in the system.

Acceptance Criteria:

The device must be able to reach +/- 1 degrees of its desired position.

Required Materials:

- Calibration Device
- Computer to run calibration software
- Protractor with hanging plumb bob

Testing Procedure:

1. Turn on the device and let it run that startup homing sequence
2. Measure the initial angle of the drone plate in the tertiary and secondary axis direction by placing the protractor flush against the bottom of the plate and reading the angle of the hanging plumb bob
3. Run step sequence for the motors to rotate a desired angular displacement
4. Measure the resulting angle of the drone plate with the same procedure
5. Repeat this cycle for both the tertiary and secondary axes, and vary the desired angular change

Data:

Tertiary Axis			Secondary Axis		
Initial Angle	Desired Displacement	Resulting Angle	Initial Angle	Desired Displacement	Resulting Angle

PASS FAIL

Test #2: Procedure Step

Description of Test:

The purpose of this test is to determine the number of steps it takes to set up the device to begin the drone calibration

Acceptance Criteria:

The number of steps required to run the device must equal five or less.

Required Materials:

- Calibration Device
- Calibration User Manual
- Inspired Flight Drone

Testing Procedure:

1. Have someone act as an Inspired Flight employee who will be loading the drone into the calibration device.
2. Record the number of steps they are required to follow as they load the drone and start the device as stated in the user manual

Data:

Number of Steps Required	

PASS FAIL

Test #3: Time to Calibrate

Description of Test:

Determine if the run time of the calibration process is within specification.

Acceptance Criteria:

The time of calibration must take less than 10 minutes.

Required Materials:

- Drone
- Calibration Device
- Stopwatch/Timer

Testing Protocol:

1. Load the drone into the calibration device
2. Turn on power to the device
3. Start the calibration process and timer

Time to Calibrate	

PASS FAIL

Test #4: Compatibility Test

Description of Test:

Determine if the device is compatible with future iterations of the drone.

Acceptance Criteria:

The calibration device must have an area that is modular and easily swapped with compatible hardware to ensure future compatibility.

Required Materials:

- Drone
- Calibration Device
- Tape measure

Testing Protocol:

1. Measure the diagonal of drone
2. Measure the secondary ring of the calibration device and see if the drone can fit within the secondary ring
3. Test to see if the Tertiary surface is easily removable and replaceable

Data:

Diagonal of Drone	
Diagonal of Secondary	

PASS FAIL

Test #5: Budget Value

Description of Test:

Determine if the device is within budget.

Acceptance Criteria:

The sum of materials must be equal to or less than $4000.

Required Materials:

- Bill of Materials

Testing Protocol:

1. Check the bill of materials to find total final cost.

PASS FAIL

Test #6: Outdoor Calibration

Description of Test:

Confirm that the system can be used outside.

Acceptance Criteria:

Pass/Fail

Required Materials:

- Fully assembled calibration system
- Drone for calibration
- Generator/power supply
- Two people minimum to carry system

Testing Protocol:

1. Carry drone outside to testing field
2. Supply power to calibration system
3. Complete calibration process

Safety Hazard:

1. Do not attempt to run calibration system in the rain
2. Wear closed-toed shoes when carrying system outside
3. Be aware of all tripping hazards in carrying path

Data:

Did the calibration system perform the necessary tasks outdoors? (Y/N) (Pass/Fail)

PASS FAIL

Test #7: Reliability

Description of Test:

Determine percentage of successful calibrations

Acceptance Criteria:

The percentage of calibration success must be greater than 99%

The percentage of flight success must be greater than 99%

Required Materials:

1. Fully assembled calibration system
2. Drone for calibration
3. Calibration operator

Testing Protocol:

1. Complete 100 calibrations over the course of multiple testing days
2. Record calibration data in table below
3. Calculate percentage of successful calibrations
4. Fly calibrated drone
5. Calculate percentage of successful flights

Data:

Calibration #	Date/Time	Calibration Success (Y/N)	Flight Success (Y/N)

PASS FAIL

Test #8: Measure the base

Description of Test:

Confirm that the base of the device fits in a 4' by 4' cube

Acceptance Criteria:

Base fits within a 4' by 4' box.

Required Materials:

1. Drone Calibrator
2. Tape Measure

Testing Protocol:

1. Use a tape measure to measure the base of the drone calibrator

PASS FAIL

Test #9: Critical Component Structural Testing

Description of Test:

Testing of critical components to confirm that they can withstand 1000 cycles under normal stress conditions.

Acceptance Criteria:

The critical components must not break when run for 1000 cycles.

Required Materials:

1. Drone Calibrator
2. Testing code
3. Observation notebook

Testing Protocol:

1. Use testing code to run the calibrator through 1000 cycles
2. If there are any problems during the testing, record the results in the notebook

PASS FAIL

<u>Test #10: Weight the Device</u>

Description of Test:

Confirm that device does not weigh more than fifty pounds.

Acceptance Criteria:

Weight of the device must be less than 50lb.

Required Materials:

1. Drone Calibrator
2. Scale

Testing Protocol:

1. Put the drone calibrator on the scale
2. Read and record the weight

PASS FAIL

Test #11: Fits Through a Door

Description of Test:

Confirm that the fully assembled drone calibrator can fit through a 32” door frame.

Acceptance Criteria:

The device must fit through a door frame without interference.

Required Materials:

- 32” door frame
- Fully assembled drone calibrator
- Tape measure

Testing Protocol:

1. Measure the door frame to confirm that the clearance is 32” from the furthest protruding parts of the frame or door. Note: from 30” to 32” is acceptable, to account for the door being partially in the way.
2. Lift the drone calibrator by the base (with two people optimally).
3. Carry the drone calibrator through doorway. Avoid contact with the door frame.

PASS FAIL

Test #12: Standard Parts

Description of Test:

Check that custom parts were required where used, thereby showing that standard parts were used when possible.

Acceptance Criteria:

No parts on the device can be replaced by standard parts.

Required Materials:

- Bill of Materials
- Assembly Drawings
- Manufactured Part Drawings
- Final Report

Testing Protocol:

1. Review the Bill of Materials for standard parts.
2. For custom machined parts, examine the design requirements and compare to existing off the shelf parts.
3. For custom parts with off the shelf alternatives, compare value using price and manufacturing time.
4. Determine if the standard part would have been a better option.

Data:

Part	Alternative	Improvement?

PASS FAIL

Test #13: Uses Wall Power

Description of Test:

Confirm that the drone calibrator uses wall power.

Acceptance Criteria:

The device runs using a standard wall socket plug.

Required Materials:

- Drone Calibrator
- Wall Outlet

Testing Protocol:

1. Plug the drone calibrator into the wall outlet and confirm that it runs.

PASS FAIL

Appendix R: Risk Assessment

Risk Level Report

Application:	RIA Baseline Risk Assessment	Analyst Name(s):	Geof F, , Kathy U, Rebecca T
Description:	Waterjet cutter operation preparing floor mats for installation	Company:	RIA Training
Product Identifier:		Facility Location:	Anywhere, USA
Assessment Type:	Detailed		
Limits:	This is a training exercise, and is not meant to be a finished risk assessment for any particular installation.		
Sources:	ANSI RIA R15 306-2016, ISO 10218		
Risk Scoring System:	ANSI RIA R15.306-2015		

Guide sentence: When doing [task], the [user] could be injured by the [hazard] due to the [failure mode].

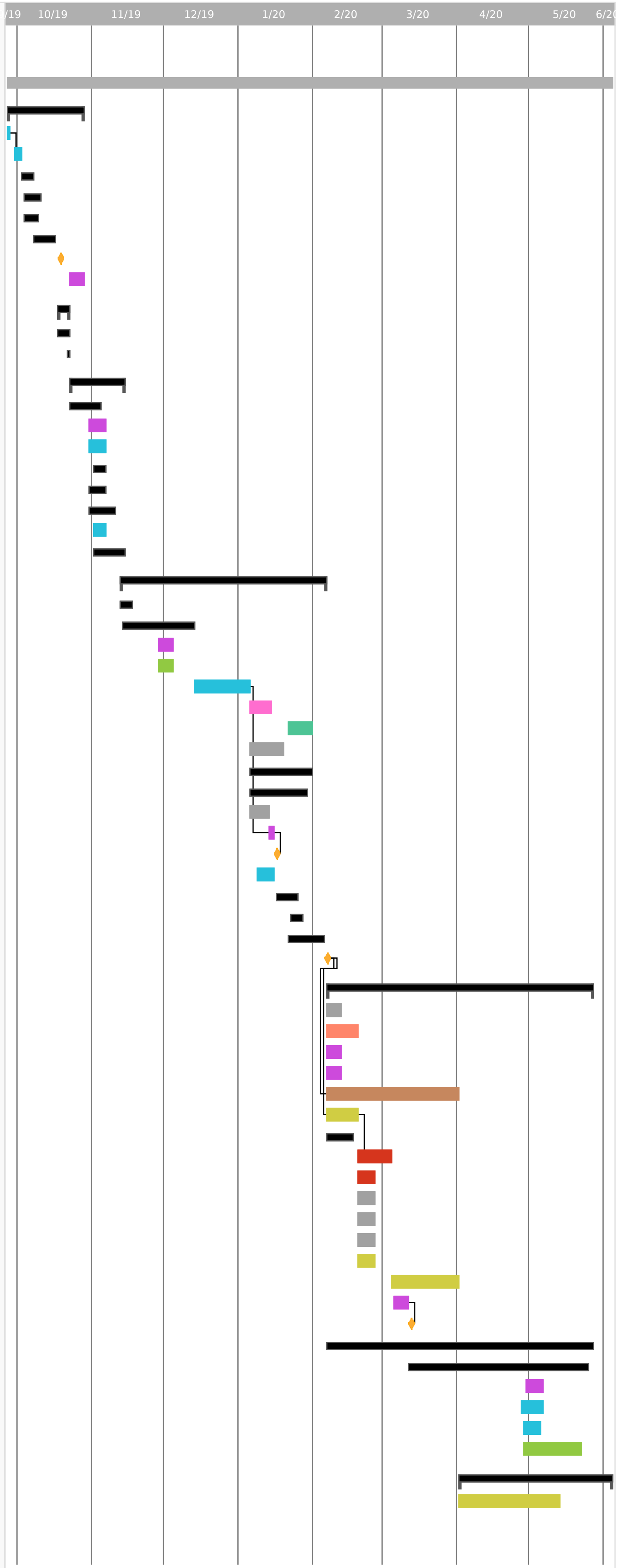
Final Assessment Severity Exposure Avoidance	Risk Level	Item Id	Sub-process / User / Task	Hazard / Failure Mode	Risk Reduction Methods /Control System	Initial Assessment Severity Exposure Avoidance	Risk Level	Status / Responsible /Comments /Reference
S3 Serious E1 Low Exposure A1 Likely	High	1-1-1-1	Loading System operator load / unload materials	electrical / electronic : unexpected start up / motion Start button is accidently pressed	special procedures	S3 Serious E1 Low Exposure A2 Not Likely	High	Action Item [2/12/2020] Jackie
S3 Serious E1 Low Exposure A1 Likely	High	1-1-2-1	Loading System operator misuse - (add description)	slips / trips / falls : falling material / object Tripping over wires when near device	safety mats / contact strip	S3 Serious E1 Low Exposure A1 Likely	High	Action Item [2/12/2020] Zach
S3 Serious E1 Low Exposure A1 Likely	High	4-1-1-1	Electrical System operator misuse - (add description)	electrical / electronic : liquid spills on electronics liquids placed near device	warning label(s)	S3 Serious E1 Low Exposure A1 Likely	High	Action Item [2/12/2020] Ryan
S3 Serious E1 Low Exposure A1 Likely	High	4-2-1-1	Electrical System passer by / non-user work next to / near machinery	electrical / electronic : energized equipment / live parts	regularly inspect electrical connections to ensure proper grounding/connections	S3 Serious E2 High Exposure A2 Not Likely	High	On-going [Daily] Tyler
S3 Serious E1 Low Exposure A1 Likely	High	4-2-1-2	Electrical System passer by / non-user work next to / near machinery	electrical / electronic : water / wet locations water spilled near device	warning label(s)	S3 Serious E1 Low Exposure A1 Likely	High	Action Item [2/12/2020] Ryan
S3 Serious E1 Low Exposure A1 Likely	High	4-2-1-3	Electrical System passer by / non-user work next to / near machinery	slips / trips / falls : trip trips over electrical wires near work station	safety mats / contact strip	S3 Serious E2 High Exposure A2 Not Likely	High	Action Item [2/12/2020] Zach

Final Assessment Severity Exposure Avoidance	Risk Level	Item Id	Sub-process / User / Task	Hazard / Failure Mode	Risk Reduction Methods /Control System	Initial Assessment Severity Exposure Avoidance	Risk Level	Status / Responsible /Comments /Reference
S3 Serious E1 Low Exposure A1 Likely	High	4-2-2-1	Electrical System passer by / non-user walk near machinery	slips / trips / falls : trip trips over electrical wires in walkway	safety mats / contact strip	S3 Serious E1 Low Exposure A1 Likely	High	Action Item [2/12/2020] Zach
S2 Moderate E1 Low Exposure A2 Not Likely	Medium	1-1-2-3	Loading System operator misuse - (add description)	pinch points : between drone and drone mounting plate hands are pinched between drone and plate		S2 Moderate E1 Low Exposure A2 Not Likely	Medium	
S2 Moderate E1 Low Exposure A3 Not Possible	Medium	1-1-2-4	Loading System operator misuse - (add description)	struck by/impact : drone falls out of system operator drops drone when loading/unloading		S2 Moderate E1 Low Exposure A3 Not Possible	Medium	
S2 Moderate E1 Low Exposure A2 Not Likely	Medium	2-1-1-1	Frame passer by / non-user work next to / near machinery	electrical / electronic : unexpected start up / motion unaware device is starting soon		S2 Moderate E1 Low Exposure A2 Not Likely	Medium	
S2 Moderate E1 Low Exposure A1 Likely	Medium	2-1-2-1	Frame passer by / non-user walk near machinery	electrical / electronic : unexpected start up / motion person is unaware machine is about to start		S2 Moderate E1 Low Exposure A1 Likely	Medium	
S2 Moderate E1 Low Exposure A1 Likely	Medium	3-1-2-1	Computer Interface operator misuse - (add description)	improper procedure : wrong button order instructions not clearly stated		S2 Moderate E1 Low Exposure A1 Likely	Medium	
S1 Minor E2 High Exposure A3 Not Possible	Low	1-1-1-2	Loading System operator load / unload materials	ergonomics / human factors : posture operator must bend and reach into device over the frame		S1 Minor E2 High Exposure A3 Not Possible	Low	
S1 Minor E2 High Exposure A3 Not Possible	Low	1-1-1-3	Loading System operator load / unload materials	ergonomics / human factors : lifting / bending / twisting must lift drone straight out of plate		S1 Minor E2 High Exposure A3 Not Possible	Low	
S1 Minor E1 Low Exposure A2 Not Likely	Low	1-1-2-5	Loading System operator misuse - (add description)	struck by/impact : hit by frame operator hits arms or legs on corners of frame		S1 Minor E1 Low Exposure A2 Not Likely	Low	

Final Assessment Severity Exposure Avoidance	Risk Level	Item Id	Sub-process / User / Task	Hazard / Failure Mode	Risk Reduction Methods /Control System	Initial Assessment Severity Exposure Avoidance	Risk Level	Status / Responsible /Comments /Reference
S3 Serious E0 Prevented A1 Likely	Low	2-1-1-2	Frame passer by / non-user work next to / near machinery	pinch points : between robot/turntable reaches hand near device while spinning	adjustable enclosures / barriers	S3 Serious E1 Low Exposure A1 Likely	High	
S3 Serious E0 Prevented A1 Likely	Low	2-1-1-3	Frame passer by / non-user work next to / near machinery	pinch points : between joints reaches hand near joints while running	adjustable enclosures / barriers	S3 Serious E1 Low Exposure A1 Likely	High	
S2 Moderate E0 Prevented A1 Likely	Low	2-1-1-4	Frame passer by / non-user work next to / near machinery	struck by/impact : hit by frame walks within circumference of rotation		S2 Moderate E0 Prevented A1 Likely	Low	
S3 Serious E0 Prevented A1 Likely	Low	2-1-2-2	Frame passer by / non-user walk near machinery	pinch points : between robot/turntable person reaches hand near device while spinning	adjustable enclosures / barriers	S3 Serious E1 Low Exposure A1 Likely	High	
S3 Serious E0 Prevented A1 Likely	Low	2-1-2-3	Frame passer by / non-user walk near machinery	pinch points : between joints reaches hand near joints while running	adjustable enclosures / barriers	S3 Serious E1 Low Exposure A1 Likely	High	
S2 Moderate E0 Prevented A1 Likely	Low	2-1-2-4	Frame passer by / non-user walk near machinery	struck by/impact : hit by frame walks within circumference of rotation		S2 Moderate E0 Prevented A1 Likely	Low	
S1 Minor E1 Low Exposure A1 Likely	Negligible	1-1-2-2	Loading System operator misuse - (add description)	wastes (Lean) : motion / moving		S1 Minor E1 Low Exposure A1 Likely	Negligible	
S1 Minor E1 Low Exposure A1 Likely	Negligible	3-1-1-1	Computer Interface operator normal operation	ergonomics / human factors : posture computer interface at awkard height for operator		S1 Minor E1 Low Exposure A1 Likely	Negligible	

Appendix S: Gantt Chart

35 - Automated Drone Calib...	start	end	100%
DEFINE	**09/27/19**	**10/28/19**	**100%**
Choose Project	09/27	09/27	100%
Meet Team	09/30	10/02	100%
Customer/Need Research	**10/03/19**	**10/07/19**	**100%**
Technical Research	**10/04/19**	**10/10/19**	**100%**
Product Research	**10/04/19**	**10/09/19**	**100%**
Write Scope of Work	**10/08/19**	**10/16/19**	**100%**
Scope of Work Milestone	10/18	10/18	100%
Revise SOW	10/23	10/28	100%
CREATE	**10/18/19**	**10/22/19**	**100%**
Brainstorming	**10/18/19**	**10/22/19**	**100%**
Function Decomposition	**10/22/19**	**10/22/19**	**100%**
EVALUATE	**10/23/19**	**11/14/19**	**100%**
Selection	**10/23/19**	**11/04/19**	**100%**
Design Hazard Checklist	10/31	11/06	100%
Update Gantt Through CDR	10/31	11/06	100%
Concept Prototype	**11/02/19**	**11/06/19**	**100%**
High Level Solidworks Model	**10/31/19**	**11/06/19**	**100%**
High Level Analysis	**10/31/19**	**11/10/19**	**100%**
Create plan for testing and analysis	11/02	11/06	100%
Preliminary Design Review Report	**11/02/19**	**11/14/19**	**100%**
SPECIFY	**11/13/19**	**02/06/20**	**100%**
Organize	**11/13/19**	**11/17/19**	**100%**
Detailed Design	**11/14/19**	**12/13/19**	**100%**
Update SOW	11/29	12/04	100%
Draft Manufacturing Plan	11/29	12/04	100%
Break	12/14	01/05	100%
Update CAD	01/06	01/14	100%
Stress Analysis	01/22	01/31	100%
Interface with QGroundControl	01/06	01/19	100%
Hardware Design	**01/06/20**	**01/31/20**	**100%**
Create Preliminary BOM	**01/06/20**	**01/29/20**	**100%**
Create Task and State Diagrams	01/06	01/13	100%
Prep for Interim Design Review	01/14	01/15	100%
Interim Design Review	01/16	01/16	100%
Plan Test	01/09	01/15	100%
Spec Out Components and Updat...	**01/17/20**	**01/25/20**	**100%**
Finish CAD	**01/23/20**	**01/27/20**	**100%**
Prep for CDR	**01/22/20**	**02/05/20**	**100%**
Critical Design Review	02/06	02/06	100%
BUILD	**02/07/20**	**05/27/20**	**100%**
Choose power supply	02/07	02/12	100%
Test/Analysis	02/07	02/19	100%
Risk Assessment	02/07	02/12	100%
Safety Plan	02/07	02/12	100%
Manufacture and Assembly	02/07	04/01	100%
Test Structural Prototype	02/07	02/19	100%
Finalize CAD	**02/07/20**	**02/17/20**	**100%**
Full Build	02/20	03/04	100%
Manufacture Custom Parts	02/20	02/26	100%
Write Step Ramp	02/20	02/26	100%
Update State Diagrams	02/20	02/26	100%
Interface with PixHawk through Tele...	02/20	02/26	100%
Testing Indoor/Outdoor	02/20	02/26	100%
Testing (Prototype)	03/05	04/01	100%
Prep for M&TR	03/06	03/11	100%
Manufacturing & Test Review	03/12	03/12	100%
Mechatronics	**02/07/20**	**05/27/20**	**100%**
Structures	**03/12/20**	**05/25/20**	**100%**
FDR Outline	04/30	05/06	100%
Finish All Plate Drawings	04/28	05/06	100%
3D Print Polycarbonate Parts	04/29	05/05	100%
Design Driver Mounting Parts	04/29	05/22	100%
TEST	**04/02/20**	**06/04/20**	**100%**
Testing and Iterative Design	04/02	05/13	100%



Final Component Selection	04/14	04/21	100%
Final Assembly	04/22	05/12	100%
EXPO website	05/13	05/25	100%
Write FDR	05/13	05/25	100%
Final Design Review	05/26	05/26	100%
Write User Manual	04/27	05/26	100%
Senior Project EXPO	05/29	05/29	100%
Celebrate	05/30	06/04	100%