

CNC Feed Drive Control

For Professor Simon Xing
California Polytechnic State University
Mechanical Engineering Department
2021

From

Juan Majano III
jmmajano@calpoly.edu

Nick De Simone
ndesimon@calpoly.edu

Ryan Funchess
rfuncses@calpoly.edu

Caleb O'Gorman
cpogorma@calpoly.edu

Samuel Wong
swong107@calpoly.edu

Statement of Disclaimer

Since this project is a result of a class assignment, it has been graded and accepted as fulfillment of the course requirements. Acceptance does not imply technical accuracy or reliability. Any use of information in this report is done at the risk of the user. These risks may include catastrophic failure of the device or infringement of patent or copyright laws. California Polytechnic State University at San Luis Obispo and its staff cannot be held liable for any use or misuse of the project.

Abstract

This document serves as the Final Design Report (FDR) for a senior project developed by our team: four senior Mechanical Engineering students and one computer engineering student at California Polytechnic State University, San Luis Obispo (Cal Poly). While the project was completed for, and sponsored by, Professor Simon Xing of Cal Poly, the remainder of the university's controls professors will be indirectly benefited from this project. Our goal was to design and implement a functional CNC Feed Drive to be used for educational demonstrations and data collection. This document discusses our early product research and benchmark goals, which established constraints for our product design, as well as identifies our design process and conclusions. Through this evaluation of the feed drive form and function, we determined optimal system components - including a DC motor with rotary encoders, a ballscrew, linear bearings, and a load table with screws for fixturing. This FDR also discusses our design progression, beginning with the structural prototype and followed by a description of the final design. This will include the manufacturing steps taken, the front-end and back-end code generated and used to control the system, and the associated user's manual. Lastly, this report will discuss the test procedures that we derived from the design verification requirements and include an overview of our test results. We conclude with our final acknowledgments, and we wanted to mention that we are extremely grateful to have worked on this project. The team has learned so much throughout the year, and we look forward to handing the prototype over to Professor Xing.

Contents

List of Tables	6
List of Figures	7
1. Introduction	9
2. Background	9
2.1 Existing Designs and Patent Research	9
2.2 Technical Information	11
2.3 Codes and Standards	12
3 Objectives	12
3.1 Customer Needs and Wants	12
3.2 Boundary Diagram	13
3.3 Quality Functional Diagram (QFD)	14
3.4 Specifications Table	14
4. Concept Design	16
4.1 Functional Decomposition and Ideation Models	16
4.2 Controlled Convergence Method	17
4.2.1 Pugh Matrices	17
4.2.2 Morphological Matrix and System Sketches	17
4.2.3 Weighted Decision Matrices	20
4.3 Concept Prototype	21
4.3.1 Concept Solid Model	22
4.3.2 Physical Prototype	23
4.4 Preliminary Design Hazards	24
4.5 Anticipated Challenges	24
5. Final Design	25
5.1 Subsystem and Component Verification	25
5.1.1 Drive Train Analysis	25
5.1.1.1 Motor	25
5.1.1.2 Ballscrew	25
5.1.2 Motion Support	26
5.1.2.1 Rail Assembly	26
5.1.2.2 Ballscrew Support Assembly	27
5.1.3 Mass Support	27
5.1.3.1 Table	27

5.1.3.2 Frame.....	28
5.1.4 Controls	30
5.1.4.1 Microcontroller and GUI.....	30
5.2 Final Design Selection	31
5.3 Safety, Maintenance, and Repair Considerations	32
5.4 Cost Analysis Summary	32
5.5 Structural Prototype.....	33
5.6 Post CDR Design Changes.....	33
6. Implementation.....	34
6.1 Part Procurement	34
6.2 Final Budget Status	35
6.3 Manufacturing/Assembly Process	35
6.4 Microcontroller Programming.....	46
6.4 User Interface Development.....	48
6.5 Challenges/Considerations	49
7. Design Verification Plan	49
7.1. Mechanical Specifications.....	49
7.2. Electromechanical Specifications	49
7.3 Test Procedures Conducted	50
7.4 Table Travel Test	51
7.5 Microcontroller Communication Test	51
7.6 Load Requirement Test	52
7.7 Table Speed Test	52
7.8 Mobility Test.....	52
7.9 Open-Loop Motor Output Test.....	52
7.10 Encoder Test.....	52
7.11 User Interface Test	53
7.12 Motion System Resolution	53
7.13 Future Testing	54
8. Project Management.....	54
9. Conclusion.....	55
References	57

List of Tables

Table 1: List of Patents for Current Models.....	10
Table 2: Current Products	11
Table 3: Customer Needs and Wants	13
Table 4: Specifications Table.....	15
Table 5: Functional Combinations from Morphological Matrix.....	18
Table 6: Proposed Design Direction	21
Table 7: Summary of Costs.....	32
Table 8: Summarized Cost of Componentry.....	35
Table 9: Testing Results Summary	50
Table 10: Measured distance from the datum after the encoder displays 5cm of actuation on the computer.	53
Table 11: Key Deliverables.....	54

List of Figures

Figure 1: This Boundary Diagram includes the focus of the project which happens to be the objects within the boundary in addition to outside influences.	14
Figure 2: Functional Decomposition Function Tree	16
Figure 3: Sketch of System 1, a “monorail” concept where the table travels along and wraps around a linear rail with corresponding grooves as shown in Section A-A above.	18
Figure 4: Sketch of System 2, in which a ball screw and servomotor is implemented.....	19
Figure 5: Sketch of System 3, where the table would be driven by a “rack and pinion” motor and rail configuration, as shown by parts (1) and (2) above.....	19
Figure 6: Sketch of System 4, which has a characteristic “suitcase” carrying feature.	20
Figure 7: Annotated CAD Assembly	22
Figure 8: Isometric View of Physical Concept Prototype.....	24
Figure 9: CAD showing both the support components.....	26
Figure 10: Finite Element Analysis of Linear Rails.	27
Figure 11: Frame isometric views with key features labeled.....	28
Figure 12: Solidworks FEA performed on the frame to ensure reasonable strength and stiffness. A maximum Von Mises stress of 383kPa and a maximum deflection of 16.1 microns is observed.	29
Figure 13: The current dummy GUI used for prototyping.....	30
Figure 14: Encoder mounting scheme from rotary encoder to ballscrew shaft.....	31
Figure 15: Full Assembly with final parts.	31
Figure 16: Schematic of the structural prototype.....	33
Figure 17: Final CAD assembly, with the implemented aluminum structure replacing the 3D Printed frame. The structure consists of a base plate, three walls, two spacers, and two legs.	34
Figure 18: Isometric view of the assembly, with the motor driving the ballscrew via flexible shaft coupling. Channel A leads from the motor driver the motor, powered by a 24V wall outlet source and controlled by a Nucleo Microcontroller.	36
Figure 19: Expanded view of Assembly.....	37
Figure 20: Ball bearing mounting configuration on the ball bearing mounting face. The ball bearing protrudes through the ball bearing mounting face and is secured with four M6 nuts and bolts.	38
Figure 21: Drawing for thrust bearing.	38
Figure 22: Expanded view for assembly.....	39
Figure 23: Top view of the system depicting the length of the ballscrew, with the ball nut connected to the table and driving on the linear rails.....	39
Figure 24: Cut out of all frame pieces.	40
Figure 25: Flattening and squaring the edges of the motor face on the Hangar mill.	41
Figure 26: Measuring and marking hole locations on the base plate and datum face using a combination square, digital calipers, and scribe.	42
Figure 27: Cutting a starting center hole of the datum face using a hole saw.	42
Figure 28: Using a bore bar on the Mustang 60 mill to cut the center hole of our “Datum Face”.	43
Figure 29: Getting help from a shop tech to find a lost set screw in the Mustang 60 mill. He found it! ...	43
Figure 30: Enclosure with Nucleo mounted.	44
Figure 31: A soldered connection between cut wires and switch terminals. We soldered one wire to the “Common” terminal and one wire to the “Normal Closed” terminal.	45

Figure 32: Limit switch mounting configuration. We mounted the limit switches at the extreme points of the table's motion on the outside of the rail and are fixed with command strips. The switches are normally closed but open the circuit upon contact with the linear bearing.	46
Figure 33: GUI as it will appear in the Controls Lab.....	48

1. Introduction

Professor Simon Xing, a Cal Poly controls professor, desires a tool to demonstrate a CNC feed drive control system to his lab students. The “Motomatic Servo Control” lab is currently run in all controls classes, however this lab has been around for over a decade and has become antiquated; hence the need for a new product to replace it. Most feed drives are comprised of a linear stage-ball screw combination and are readily available in industry at a variety of quality levels and price points. However, there are no existing products that are both within the university’s budget and educationally effective. We – Caleb O’Gorman, Juan Majano, Nick De Simone, and Ryan Funchess – have answered the call and are looking to create a tool that will meet all of Professor Xing’s needs and will be used for years to come within Cal Poly’s engineering department.

The following sections begin by highlighting in detail the existing products we have discovered in our preliminary research, including contemporary patents and processes. Afterwards, Professor Xing’s problem is more explicitly defined with the help of a House of Quality (see Appendix A.2). We have defined each of these background research and problem definition sections further in Chapters 2 and 3 of this report – titled “Background” and “Objectives”, respectively. Much of this project initialization was refined through ideation, matrix analysis, and prototyping, all of which may be observed in Chapter 4 “Concept Design”, whereupon our team landed on a final direction for the product. Following design selection, the “Final Design” chapter introduces our structural prototype, along with a presentation of our manufacturing and design verification plan – Chapters 5, 6, and 7, respectively. Our indented Bill of Materials (see Appendix A.11), drawing package (see Appendix A.15), and failure modes and effects analysis (see Appendix A.12) are also presented and evaluated as appendices. Lastly, we discuss our project management in Chapter 8, explicitly outlining the year’s schedule in our Gantt Chart (see Appendix A.3).

2. Background

We had the pleasure of first meeting with our sponsor Dr. Xing on September 25th, 2020. We performed brief introductions, went over our version of the project objective, and established a bi-weekly meeting time. In the following meeting we went through our problem definition with Dr. Xing and conducted further inquiry as to his wants and needs for the project moving forward. He dictated that he needs a CNC feed drive system that is cost effective, educational, and built to last a long time. In addition, we gained insight about what would be desired from consulting with Professor Birdsong and Professor Hemanth Porumamilla.

2.1 Existing Designs and Patent Research

In performing our patent research, we examined existing products within an actual CNC machine to develop an understanding of system functions. After taking the time to learn about components of a CNC feed drive, we began our patent research by looking at linear stage applications. We learned that a feed drive consists of a motor driven by an electronic controller, a feed system driven by a motor, as well as a housing for all components. In addition, the feed motor provides constant torque at steady state and allows for high precision actuation of a mass along an axis. According to an article from Tooling World [19] some qualities of a CNC feed drive include:

- Maximum motor speeds of 3000 rpm
- Low electrical and mechanical time constants
- Low armature or rotor inertia
- High peak torque for quick responses

We deduced that industry standard linear stages, such as from Newport and Aerotech, have a standard minimum cost of \$5,000 depending on application.

Our goal is to mitigate cost and maintain quality by looking at each component individually. Table 1 displays patent descriptions of products related to our project.

Table 1: List of Patents for Current Models

Patent Name	Patent Number	Key Highlights
Automatically guided tools ^[1]	US 10067495 B2	<ul style="list-style-type: none"> • Method and tool for guiding a tool during its use based on its location relative to the material being worked on. • Introduces the idea of a rig or frame with <u>stage</u> that can be positioned on the surface of a piece of material.
Servomotor controller for controlling a servo motor designed to drive the feed axis in a machine tool ^[2]	DE 102013103341 B4	<ul style="list-style-type: none"> • A servo motor control apparatus which performs feedback control of a servo motor adapted to drive a feed axis
High-speed high precision servo linear motor sliding table ^[3]	CN 10184431 4A	<ul style="list-style-type: none"> • The servo linear motor sliding table can be applied to high precision feeding mechanisms of numerical control turning machines.
Exposure apparatus movable body apparatus, flat panel display manufacturing method, and device manufacturing ^[4]	KR 101911717 B1	<ul style="list-style-type: none"> • The object holding member holding the object together with the first moving body is driven in the direction parallel to the first direction.
Power Tool with a linear motor ^[5]	US 6705408 B2	<ul style="list-style-type: none"> • Linear motor which includes a movable element having a tool on it on end.

We conducted our patent search by using Google Patents and identifying keywords which are applicable to our project (i.e. “servo system components” or “feed drive applications”). We are aware that similar products exist, however, our main goal is to make a feed drive as affordable and as classroom ready as possible, which proved to not be a readily available application of such product. Please see Table 2 for a list of similar servo system products and their key descriptions, which we have and will use to inform our own design moving forward.

Table 2: Current Products

Company	Product Type	Key Highlights
Aerotech ^[6]	Linear Stage	<ul style="list-style-type: none"> High precision (5nm) actuation of a table along a single axis. Long travel distance, high resolution and speed, high price. Zero backlash, zero friction. Uses a recirculating linear guide bearing system with encoders
Newport ^[7]	Linear Stage	<ul style="list-style-type: none"> High precision(10nm) linear stage actuation. High load, travel, repeatability models available. High price. Options with Piezo, stepper, DC, brushless DC, and linear motors. All closed loop options use encoders.
Misumi ^[8]	Linear Stage	<ul style="list-style-type: none"> Lower precision stage (microns) ballscrew actuation at required table length. Variety of motor options
Grainger ^[9]	Linear Actuator	<ul style="list-style-type: none"> High load linear actuation using gears, including worm gears for high load Short stroke length, no feedback, low precision
Alum-a-Lift ^[10]	Custom Lifts	<ul style="list-style-type: none"> Custom actuators used to move high load in multiple axes. Worked with many companies in industry.
McMaster-Carr ^[11]	Hydraulic Actuator	<ul style="list-style-type: none"> Hydraulic positioners that can produce very high force with lower precision than ballscrews and gears

2.2 Technical Information

We sought further understanding of our project scope through research into scholarly journals. In this research, we were informed as to the complexity of servo systems, such as by Kaan Erkorkmaz and Yusuf Altintas in their International Journal of Machine Tools and Manufacture journal article, which presented methods for identifying the dynamic parameters and frictional characteristics of machine tool drives. In addition, they demonstrated the overall axis model used in designing a high-speed feed drive control system.^[12]

The second journal written by Kuldeep Verma and R.M Belokar investigates the performance and positioning accuracy of numerical controlled (CNC) feed drive systems while using a ball screw-based preloading impact factor.^[13] From this the main thing we learned was that the optimum pre-loading value had been determined by analyzing the available ranges from there it was proposed that those optimal results have been achieved at 5 percent of dynamic load rating. This will help us when it comes to controlling the movement of the table.

The third journal written by Breaz Radu-Eugen and team goes into the tuning process for a CNC feed drive which will serve a purpose later in the project life.^[14] This journal a general insight into how the time values having to be changed by the user to preserve positioning and contouring accuracy of the machine. This will be very helpful when it comes to tuning the motor that we decide to move forward with.

The same group of authors that wrote about the tuning process published another journal article where they focused on the dynamic behavior using MATLAB and Simulink. They then used these analyses to give suggestions to optimize the design and improve the system's dynamic behavior.^[15] Looking at a block diagram put the process in relatable terms as we all have experience modeling controls systems in Simulink. It gave us a better understanding of how a closed-loop model would work.

One of the things we considered during this early design phase was possible alternatives for the motion actuation. R.J. Wai published a journal entry about using induction in combination with a servomotor to

get a more precise motion.^[17] A notable downside of this system is the high cost and magnetic interference. They mention that an adaptive sliding-mode control system is used to control the position of an induction servomotor drive. This will help us when it comes to moving the table over the full range of motion.

For safety and durability matters, we also investigated the vibration patterns of a typical ballscrew feed drive. Ya Zhang and a team of researchers conducted an in-depth analysis of the natural frequencies and transient response of a ball screw drive using a 6 Degree of Freedom model.^[18]

Additionally, we sought out information from various sources relating to high precision actuation. Tooling World provided specifications based on current CNC Feed Drives, such as typical degree of precision, speeds, and the necessity of feedback in the control system^[19]. Newport's Technical Notes have also proven to be a valuable resource. Their technical note "Stage Components Considerations" details the positives and negatives of different modes of mechanical actuation, motor selection, as well as materials choice.^[20]

We are hoping that these journals and online manufacturer publications will help to support our project by delivering insight into the accomplishments in our project area, as well as components that need research and development.

2.3 Codes and Standards

Industry codes and standards provided our team insight as to how we may need to conduct the manufacturing, installation, and eventual operation of our servo system. We encountered the NAICS 533531 Electrical Equipment Manufacturing Code^[21] which covers the manufacturing of an electric motor like our product's servo motor, as well as the Occupational Safety and Health Administration's Electronic Code of Federal Regulations (eCFR), which informs into standard safety practices for the installation and operation of our electromechanical system^[22].

3 Objectives

After performing research related to current products and technologies, we focused in on project specifics relating to customer desires, design criteria, and engineering specifications.

3.1 Customer Needs and Wants

After discussion with both our sponsor and other Cal Poly Controls Professors, including Dr. Hemanth Porumamilla, Dr. Charles Birdsong, and Dr. John Ridgely who is leading the design of the new ME 418 Controls Lab, we drafted a problem statement and identified custom needs and wants. Our problem statement stemmed from the idea of the system as an educational tool for future controls students, rather than as a simple linear actuation device. Our problem statement is detailed here:

"Dr. Xing and the Mechanical Engineering department need a cost-effective way to physically demonstrate concepts related to motor controlled linear actuation in the form of a CNC feed drive to their control's students in order to augment the controls laboratory experience and allow students to learn by doing."

Based on this problem statement, we communicated with our sponsor about his needs and wants. From the categories listed in the provided template, we organized customer desires as shown in Table 3.

Table 3: Customer Needs and Wants

Category	Needs and Wants
Geometry	Travel distance of at least 30cm.
Motion/Kinematics	Precise linear actuation along a single axis.
Forces/Torques	Motor torque transports 5kg payload along single axis.
Material	Rails must handle payload transport; other materials must minimize degradation over time
Signals	Digital Inputs
Safety	Direct protection (eyeglasses), operational safety (pinch points; do not put hand in track or near motor; do not be in contact with system during operation; refrain from having hair, jewelry, clothing, and etc. near mechanism). Use of limit and kill switches.
Human Factors/Ergonomics	Visible components, clearness of layout
Quality Control	Possibilities of testing and measuring, application of special regulations and standards
Assembly	Easy to assemble and modular in nature. Special regulations, installation, siting, foundations. Use of standard parts to decrease cost.
Operation	Easy to work with for students with standard software (Simulink, MicroPython).
Costs	As low cost as possible to stay within our budget.
Schedules	End date of development, project planning and control, delivery date.
Portability	Dr. Xing desires a small, yet fully functional, model to be shared amongst controls labs. Dr. Xing prefers a transportable model. Consider microcontroller to drive servo system and accomplish portability
Durability	Model must be able to withstand handling by multiple labs and repeated usages across school years, designed for fatigue.

These needs and wants were selected for the purpose of a system that can precisely actuate a payload along a single axis, as well as for ease of work for students. For ease of work, the system must be compatible with software that students are familiar with, such as MATLAB and MicroPython, and a clear layout must be selected during system design. The use of digital inputs, rather than the analog inputs that are currently used on the “Motomatic” lab experiment, will allow for ease of data collection and communication between the encoders and the lab computers. Professor Xing ultimately desires a portable system for increased flexibility and use between controls labs.

The motorized system must be able to allow translation of a payload mass at reasonable speed and with reasonable accuracy to ensure timely, accurate data collection and laboratory practices. Strength and stiffness must be considered during design to ensure a CNC Feed Drive model that is durable in the long run. In addition to machine safety, the safety of humans working with the machinery must be optimized to minimize pinch points and hazards relating to rotating components.

3.2 Boundary Diagram

Figure 1 below addresses our project scope as a boundary diagram. This sketch includes our sponsor and the students who will be benefitting the most from this project. They are presented outside of the boundary

because they are out the scope when it comes to producing the product however, they are still important, nonetheless. What is included inside of the scope are some the major components needed to achieve project requirements these needs/wants will be discussed momentarily.

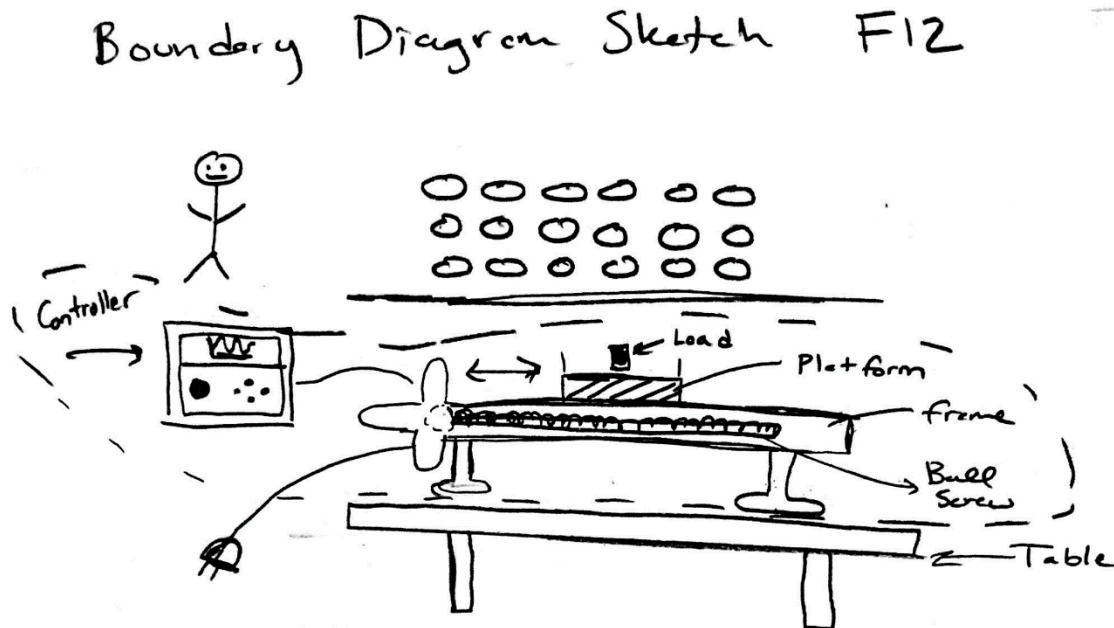


Figure 1: This Boundary Diagram includes the focus of the project which happens to be the objects within the boundary in addition to outside influences.

3.3 Quality Functional Diagram (QFD)

After discussion of Customer Needs and Wants, we used the Quality Functional Deployment method (QFD) in order to match customer requirements to engineering specifications, gain a better understanding of the problem that we are trying to solve, see which of our criteria matters most, and observe what we can bring to the table that our competitors have not already. We identified the main stakeholders and weighted the wants and needs of each stakeholder with Pairwise Comparison Charts, as attached in Appendix A.1. We observed our competition on the right side of the House of Quality and we found that although linear stages exist that are designed for high load and precision, no model competitor that we observed had a design for educational purposes. In addition, linear translation devices for the specified travel distance are more expensive than our budget. Finally, we came up with engineering specifications that could be used to quantify and fulfill each need and want and we compared each specification against each other in the roof to observe the relationship between these specifications. The House of Quality that we developed can be observed in Appendix A.2.

3.4 Specifications Table

Based on our QFD, we developed the Specifications Table seen in Table 4. The purpose of this table is to assign measurable targets to the engineering specifications that will be used to fulfill our customers' needs and wants. Each of the ten specifications can be measured with a target value that stands as either a maximum, minimum, or target value for the requirement. The risk of each specification is also laid out as Low(L), Medium(M), or High (H). Finally, the Compliance column dictates how each of these

specifications will be verified. The verification methods used for these specifications include Analysis (A), Testing (T), Inspection (I), and Similarity to Existing Designs (S).

Table 4: Specifications Table

Spec. #	Parameter Description	Requirement or Target	Tolerance	Risk	Compliance
1	Table Travel Distance	30 cm	Target	L	I
2	Load Requirement	5kg Payload	Min	H	A, T
3	Mobility of Unit	System Mass 50lb	Max	M	I
4	Intuitive	Must be able to be understood in 30 min	Max	H	T
5	Low Travel Tolerances	50 Microns	Min	M	A, T, S
6	Low Cost	\$500 System Cost	Max	H	A
7	Drop Height Durable	Survive drop height of 3.5 ft	Min	M	A
8	# of Cycles	60 hrs/yr	Min	M	A
9	Power	200 W	Max	L	A, I, T
10	Speed	1 cm/s	Min	M	A, T

*Note: H=High; M=Medium; L=Low; A=Analysis; T=Testing; I=Inspection; S=Similarity

Each of these specifications is described in further detail here:

1. **Travel Distance Requirement:** The payload that is mounted onto the linear stage should be able to linearly traverse a 30cm distance. This will be tested by inspection of sizes when we are designing the system and ordering parts.
2. **Load Requirement:** This system will actuate payloads with maximum mass greater than 5kg. This will be tested through both analysis of the power transmission to the mass from motor to payload, as well as testing with payloads of variable mass before commission.
3. **Mobility of Unit:** System mass excluding the payload shall be under 50lbs in order to maximize mobility of the unit. This will be tested by inspection. We will keep track of the mass of all parts and add the mass of all components to calculate a lump sum mass for the system. We will additionally weigh the system prior to commission.
4. **Intuitive:** The system must be intuitive enough to be understood by users within a 30-minute timespan. The intuitive aspect of the design will be tested with focus groups of students that have both passed Controls, as well as those who have fulfilled the prerequisites and will take the class in the future.
5. **Low Travel Tolerances:** The payload position must be certain along the axis of travel to within 50 microns. This will require analysis based on the motor, gearbox, and actuation system that we choose to use, as well as testing in the lab to ensure these tolerances can be satisfied. Due to the similarity of this technology to other linear actuators, we expect this requirement to be satisfied with standard parts.
6. **Low Cost:** Overall system cost requires a sponsor cost of no more than \$500 to Dr. Xing. Cost will be verified through analysis by keeping a Bill of Materials (BOM) and vendor prices as we receive quotes for our hardware.
7. **Drop Height Verified:** This system must be designed to survive a drop height of 3.5 ft in the case misuse. We will conduct a shock load analysis for the system.

8. **# of Cycles:** This system must be able to run 60 hrs/year, which was based on the duration that this system will run in lab, average amount of labs per week, and average academic weeks per year. While testing is not possible given current time and resources, we will perform a fatigue analysis on rotating components to design for long life.
9. **Power:** The system will draw a maximum of 200 W of power. We will inspect motor power consumption while purchasing parts, perform analysis if required power draw will meet other system requirements, and test motor power draw in the lab with the use of a multimeter for measurement of voltage and current.
10. **Speed:** The payload shall actuate at a minimum top speed of 1 cm/s. This will be verified through power transmission analysis and testing in lab.

4. Concept Design

After performing extensive background research and firmly establishing the needs of our customers, we set forth on the process of ideating. The ideation process consisted of first suggesting rudimentary solutions for each of our 6 functions, followed by a method of controlled convergence in which we eliminated the less desirable ideas and methodically arrived at our concept prototype, which represents our proposed design direction.

4.1. Functional Decomposition and Ideation Models

The ideation process began with functional decomposition, where we defined the CNC feed drive's overall operation with six independent functions. Naturally, the six functions housed sub functionalities to provide a comprehensive definition of the feed drive product; please see Figure 2 for our Functional Decomposition Function Tree. Note that six major functions are identified in the first level of the function tree; their subfunctions are defined in the second level and any third level of functionality is meant to further clarify its parent functions.

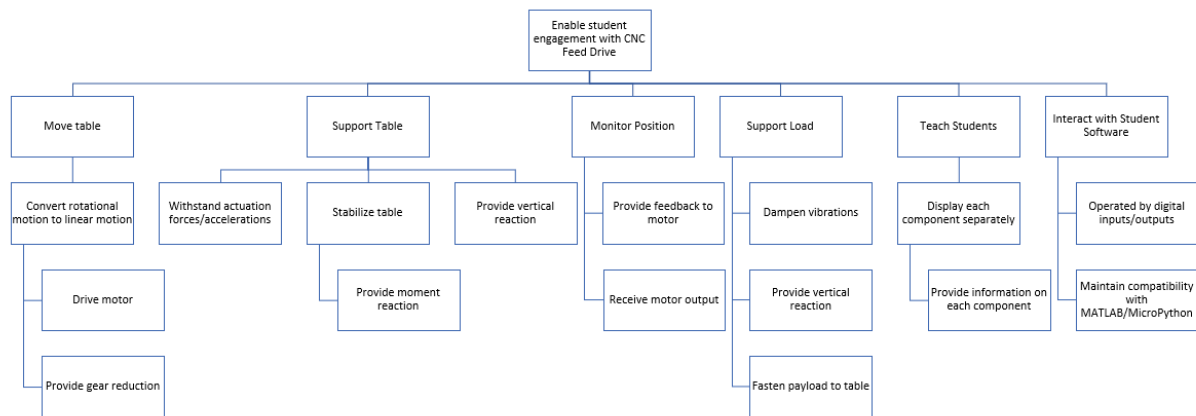


Figure 2: Functional Decomposition Function Tree

The six main functions of Figure 2 informed subsequent idea generation. These task descriptions (“move table”, “support table”, etc.) were used to prompt mechanical means for their execution, and our initial approach for product development was quantitative and unrefined. We focused on generating ideas for product components in large numbers, rather than limiting or criticizing suggestions based on their feasibility. Our team held virtual collaboration on Google’s “Jamboard” to record any proposed solutions

– again regardless of their practicality. For example, the function header “Move Table” prompted responses that ranged from “rack and pinion” to “blow on it really hard with a fan” we had to come with as many ideas as possible regardless of silliness. Please see published photos of our “Jamboard” ideation results in Appendix A.4 for a visual of all our generated ideas.

The team’s brainstorm sessions led into physical conceptualization of function mechanisms through the construction of ideation models. Each member used rudimentary materials, i.e. cardboard and hot glue, to test basic practicality of our generated ideas in rapid fashion (see Appendix A.5 for photos of ideation models). This practice allowed for insight into component feasibility, informing a more tangible understanding of system interactions and of the parts the system will need to operate correctly. Out of this ideation, we moved into evaluation and synthesis of our function idea lists and rudimentary models.

4.2. Controlled Convergence Method

We refined the ideas and models generated during our ideation phase through “controlled convergence”. Here, we looked to filter design and component proposals by evaluating them for feasibility and efficacy. Many of our Google’s “Jamboard” suggestions were eliminated to produce a list of reasonable design considerations for our main functions. These condensed function lists allowed us to create Pugh matrices for each of the functions, and our team evaluated the components against one another using the matrices, ultimately producing the most suitable options. Subsequent convergence involved further system evaluation and refinement through morphological and weighted decision matrices, each described at length in the sections below.

4.2.1 Pugh Matrices

Our Pugh matrices were constructed by using our customer needs and wants as criteria to evaluate product considerations against each other. One Pugh matrix was constructed for each system function, with the most apt idea set as a datum for comparison. Note that subsequent ideas were assigned a “+,” “-,” or “s” for their ability to meet the design criteria for better than, worse than, or the same as the datum, respectively. These symbols were counted as 1, -1, and 0, respectively, then summed to produce total values for each design idea. Low scoring ideas were eliminated while high scoring ones were carried into the next ideation phase, which involved morphological matrices and system sketches and is discussed in Section 5.2.2.

Please see Appendix A.7 for each of our five Pugh matrices: Move Table, Support Table, Monitor Position, Support Load and Portable System. The datums largely remained the best-performing system components, but we were able to glean one or two additional ideas for consideration in the final concept prototype.

4.2.2 Morphological Matrix and System Sketches

After narrowing down our list of solutions to about 5 top ideas per function, we used a morphological matrix to pair the various ideas to create 4 system-level alternatives. During functional decomposition, we had initially defined 7 main functions; however for the morphological matrix we determined that two of the functions, “Teach Students” and “Digital Inputs,” were uninfluenced by the other system components, and thus were not included. In other words, each of the solutions under these two functions could be paired just as easily with any other solution and be completely unaffected. Furthermore, many of the solutions under these functions were not mutually exclusive with each other and the system could very well be benefited by the inclusion of all of them.

After listing out the functions and their corresponding ideas, each group member selected a set of solutions, one for each function, and created a system level sketch incorporating all the ideas. The goal was to analyze how the different functions interact with each other and assess the feasibility of the different combinations. The various mixes that were derived are presented below in Table 5. The functions are organized by

columns and the system level alternatives are presented row-by-row. Scans of the drawings for each of the four concepts are shown in Figures 3-6.

Table 5:Functional Combinations from Morphological Matrix

System #	Move Table	Support Table	Monitor Position	Support Load	Make System Portable
1.	Ballscrew	Monorail	Linear Encoder	Screws	Briefcase
2.	Ballscrew	Wheels and Rail	Linear Encoder	Vice	Removable Cart
3.	Rack and Pinion	Linear Bearing	Rotational Encoders	Screws	Fixed Cart
4.	Ballscrew	Linear Bearing	Linear Encoder	L-Bracket	Suitcase

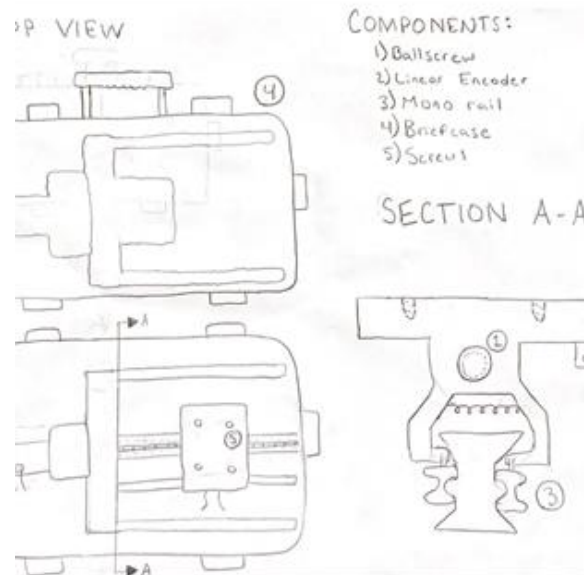


Figure 3: Sketch of System 1, a “monorail” concept where the table travels along and wraps around a linear rail with corresponding grooves as shown in Section A-A above.

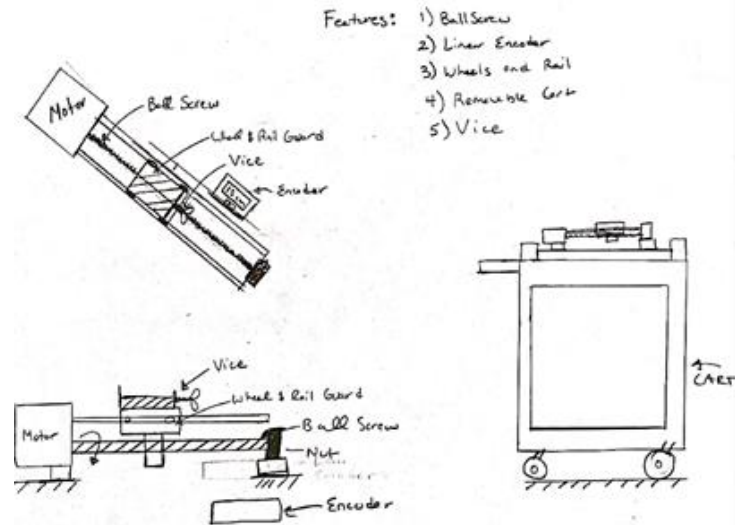


Figure 4: Sketch of System 2, in which a ball screw and servomotor is implemented.

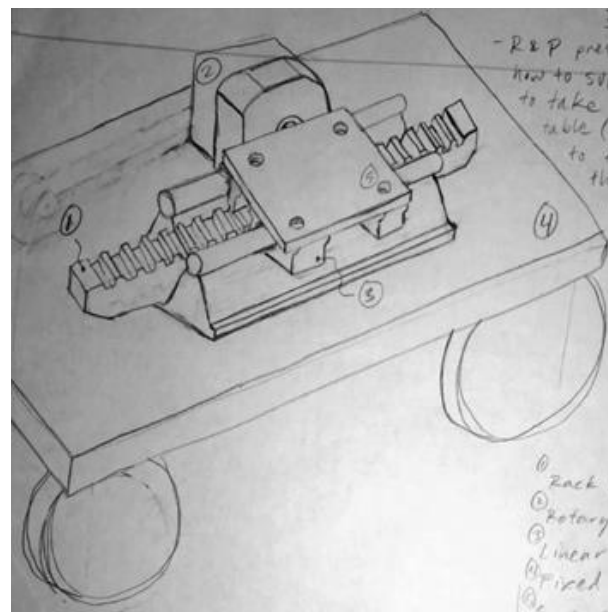


Figure 5: Sketch of System 3, where the table would be driven by a “rack and pinion” motor and rail configuration, as shown by parts (1) and (2) above.

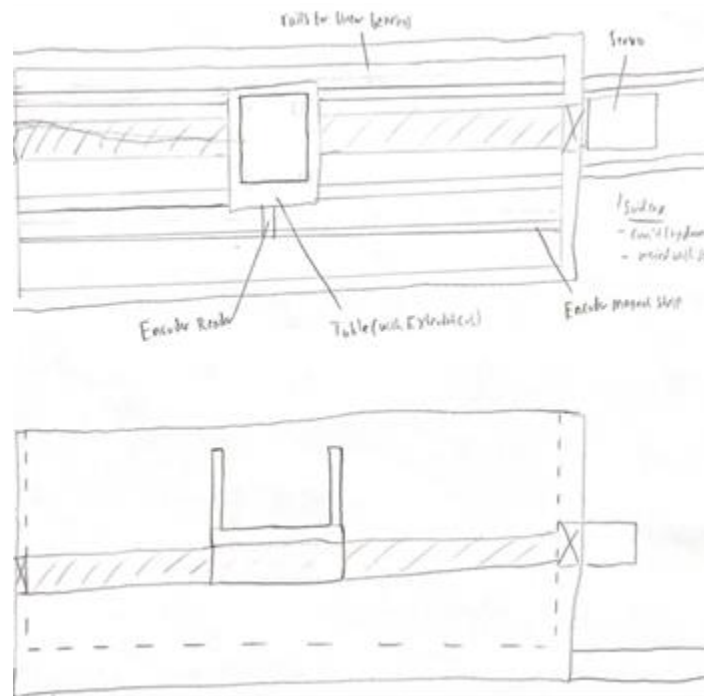


Figure 6: Sketch of System 4, which has a characteristic “suitcase” carrying feature.

The basic characteristics that each system sketch share are all integral parts of linear stages that we designated “non-negotiables,” considering our product is educational in nature and meant to resemble feed drives commonly found in industry. These attributes include a horizontal table with which to carry the mass and, except for System 2, a box-shaped enclosure in which to contain the system. System 2 explored somewhat of a unique approach to modularity in which the system was mounted on supports included on a portable cart. This idea was new to us and was included in our subsequent decision matrices.

The sketches served their purpose in helping us assess the feasibility of several of the pairings. For one, systems 1 and 4 revealed that geometrical constraints make it difficult to pair a centrally located table support, such as the monorail, with a centrally located drive, such as a ballscrew or rack and pinion. Subsequently, the impracticality of alternative drive systems acting on the system’s periphery began to steer us towards peripheral table supports such as the rail and wheels or the linear bearings. It therefore came as no surprise that these are the support mechanisms that represent industry-standard.

4.2.3 Weighted Decision Matrices

After developing four system-level alternatives, we began the process of determining the best model with which to move forward. Our initial plan was to create a weighted decision matrix with the concepts we had sketched from the morphological matrix; however, after some preliminary discussion we felt that we could get a better result if we looked beyond these and instead created a new system-level combination. We wanted our design direction to encompass all the “best” ideas for each function, assuming that they were all compatible with each other. Therefore, we decided to pursue 5 different weighted decision matrices, one for each function, to settle which were the leading design elements.

The final decision matrices are displayed in Appendix A.8. The rows are organized by our customer’s needs and wants, while the columns are arranged by the different design ideas for each function. Each cell assesses its column’s ability to address the requirement presented in its row’s heading. The cells were ranked on a

1-3 scale, 3 meaning they addressed the need very well, and 1 meaning they hardly accomplished the goal at all. Each customer requirement was given a weight based on its priority that we determined during the research phase. Note that not all customer specifications were included for each function, as not every function is meant to accomplish every need/want. The culmination of the top scoring elements for each function is presented below in Table 6.

Table 6: Proposed Design Direction

Function	Solution
Move Table	Ballscrew
Support Table	Linear Bearings
Monitor Position	Linear Encoder
Support Load	Screws
Make System Portable	Cart

For moving the table, we opted for the ballscrew method of actuation. While ballscrews oscillate at a low natural frequency at long lengths, and cost more than belts, they are high in precision, strength, and durability. In addition, they are industry standard for precision actuation over the specified travel distance, rotate with low friction, and provide a direct transmission of power from the motor's rotary motion to the linear motion of the payload. To support the table, a linear bearing was chosen. The deciding factor was its high precision actuation and educational effectiveness. We believe it will minimize friction, which will allow students to model this as a linear system. To secure the mass atop the table's surface we decided to include screws and threaded holes to fasten it. This mechanism is easy to setup and effective in the classroom laboratory setting, especially considering its low cost, strength, and portability.

To precisely locate the table's position, the linear encoder is a better option than rotational encoders. While a linear encoder costs more and requires more wiring, it is also more precise and measures the true position of the table, rather than the output position of the motor. This will contribute to the educational simplicity of the model, as it will not require a gain to convert between motor output angle and linear travel. Finally, to meet Dr. Xing's portability requirement, the clear direction seems to be wheeling the system on a small cart. While the alternatives, such as the suitcase or briefcase model, seemed both effective and stylistically desirable, they were greatly outscored in terms of cost, both fiscal and effort.

To ensure the feasibility of this proposed design direction, we conducted a preliminary numerical analysis in Appendix A.9 to estimate the amount of power required to drive the ballscrew at an assumed speed of 0.01 m/s. Using conservative estimates for frictional losses and efficiency of the motor, geartrain, and ballscrew we found the required motor power to be in the range of 2.5 W. Even had we assumed a speed of 10x this, i.e. 10 cm/s, the required power would only be 25 W. This speed would allow the table to cover the entire table distance of 30 cm in 3 s, significantly faster than what we either expect to be needed, or what we expect the ballscrew's natural frequency to permit. Even still, this estimate of 25 W is well below the rated power output of some Servo motors that we have found, which are often between 50 and 200 W.

4.3. Concept Prototype

In this section, we discuss the conceptual solid model design that we used to construct a basic system layout, as well as our physical concept prototype that was used to show the validity of our ideas in real life.

4.3.1 Concept Solid Model

After completion of the weighted decision matrix, we constructed a CAD model in Solidworks to better visualize the placement of all components for development of the system. A labeled isometric of our conceptual CAD assembly can be observed in Figure 7.

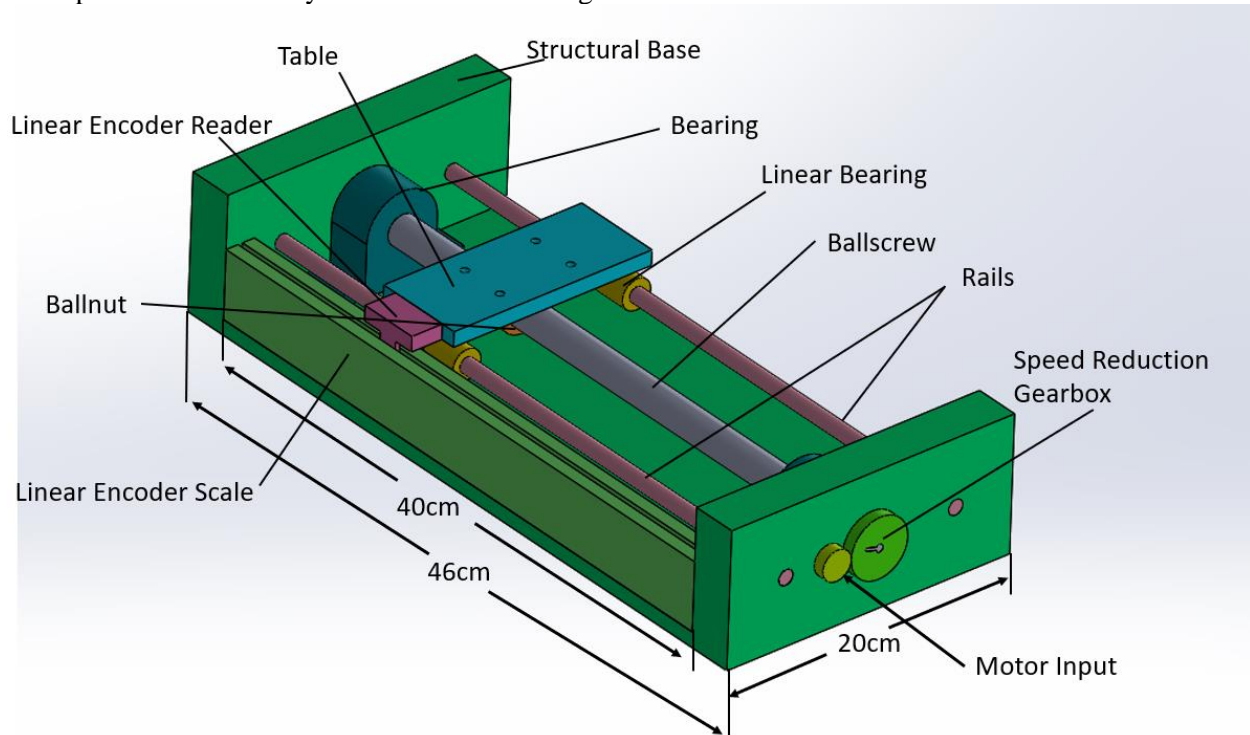


Figure 7: Annotated CAD Assembly

When designing, we started with the essentials of the power transmission from motor rotation to payload motion. Power is transmitted from the motor to the speed reduction gearbox, where torque increases proportionally to the decrease in velocity. This gear reduction will allow for higher precision by decreasing output positional uncertainty by the same amount as output velocity, in addition to increasing motor torque to allow for use of a smaller motor. Detailed gearbox design will occur during system detailed design. The output gear will be connected to the ballscrew shaft through a mating keyway and either press fit or secured with setscrews. The ballscrew will be supported by two bearings to allow for shaft rotation, while inhibiting translation. One of the bearings will be selected as a thrust bearing, which will be used to provide axial support to the ballscrew. Rotation of the ballscrew causes the internal ball bearings to circulate, which drives the ballnut longitudinally with minimal internal friction. The ballnut will directly connect to the table through either bolts or welds to ensure a rigid connection between the two components. Two linear bearings attached to the table will slide on rails parallel to the ballscrew, which both support weight and prevent tipping by adding a moment reaction. The use of a single linear bearing on each side will decrease chances of the system binding up during translation.

The mass to be mounted on the table will be a series of thin plates with 4 clearance holes to line up with the four tapped holes on the table of the CNC Feed Drive. Screws will thread into the table while clamping down the mass, which keeps a rigid connection between the table and mass while maintaining the screw in tension. This will allow students to easily stack a variable amount of masses to observe the effect of increased mass on system response.

For structural stability and transportation, the entire system will be constructed within a structural base that will provide support to the rails, bearings, microcontroller, and linear encoder reader. This structure will be sized to fit all subsystems and will likely either be 3D printed from Polylactic Acid (PLA), manufactured from sheet metal, or cast, depending on the necessary reaction forces required. A cover made of the same material will be used to cover hazardous rotating machinery when in use, as well as prevent dust and other substances from accumulating in critical spots, such as the linear encoder scale.

In order for the system to run in closed loop configuration, the linear encoder has been selected to return information regarding table positioning to the controls algorithm that we will develop in order to locate the table to a position using a PID controller. A linear encoder reader that travels with the table will work in conjunction with a linear encoder scale that runs the travel length of the table to send voltage output to the microcontroller, which will use built in board functionality to digitally measure distance. Cables will travel with the table to provide a direct connection between the linear encoder reader and the microcontroller. Using the control system that we will design; the microcontroller will incrementally determine the amount of voltage to apply to the motor driving the system. For safety, we will additionally install a limit switch and a kill switch at the limits of the table's translation in order to cut power to the system before catastrophic failure occurs in the event of a control system error or malfunctioning of the microcontroller.

4.3.2 Physical Prototype

After development of the solid model, the next step was to build a physical concept design, which would allow our team to demonstrate the physical functions of the feed drive. We took the top ideas for each function presented in Table 6 and integrated them into a physical model based off the conceptual Solidworks Model. Our team made great use of the Lego Technic pieces from the ME 329 Build Kits that were provided to us by Professor Schuster.

We began our prototype build day by cutting a cardboard frame. Next, we selected our Lego pieces, which included the axles, spacers, connectors, spur gears, and worm gears. We used joined colinear Lego worm gears to represent a ballscrew, and the table was constructed using Lego beams and joints. We punched holes in the frame to create the placement for the rails, and we actuated the table by mating a rigid spur gear from the table to the joined worm gears, such that the rotation of the worm gear caused translation of the table. A temporary cart was crafted to illustrate system portability, with four clamps used to secure the system at the corners.

From this process, we experienced firsthand that additional thought will have to be put into placement of components in the system. The Solidworks model allows for easy manipulation of the system to illustrate the interaction between system components but lacks a feel of real effects such as gravity, friction, deflection, and other imperfections. For example, our prototype forced us to mount the input gear to a bearing on the structure and ensure a small lever arm between the gear and the structure to avoid shaft deflection. At the same time, the gears were not allowed to contact the structure due to effects caused by friction, so an adequate length was found by trial and error. Due to the robust design of our ballnut and rails, space is much tighter than anticipated, and we will have to design the structure to be spacious enough to fit all necessary components, including the linear encoder cables, scale, and the microcontroller that will run the motor. We will have to design our layout such that cables do not interfere with linear translation. Additionally, the table had to be aligned precisely with the ballscrew and rails for uninterrupted linear translation to occur, so special attention must be paid to alignment and location of these cylindrical components during manufacturing. A picture of our physical prototype can be seen in Figure 8 below.

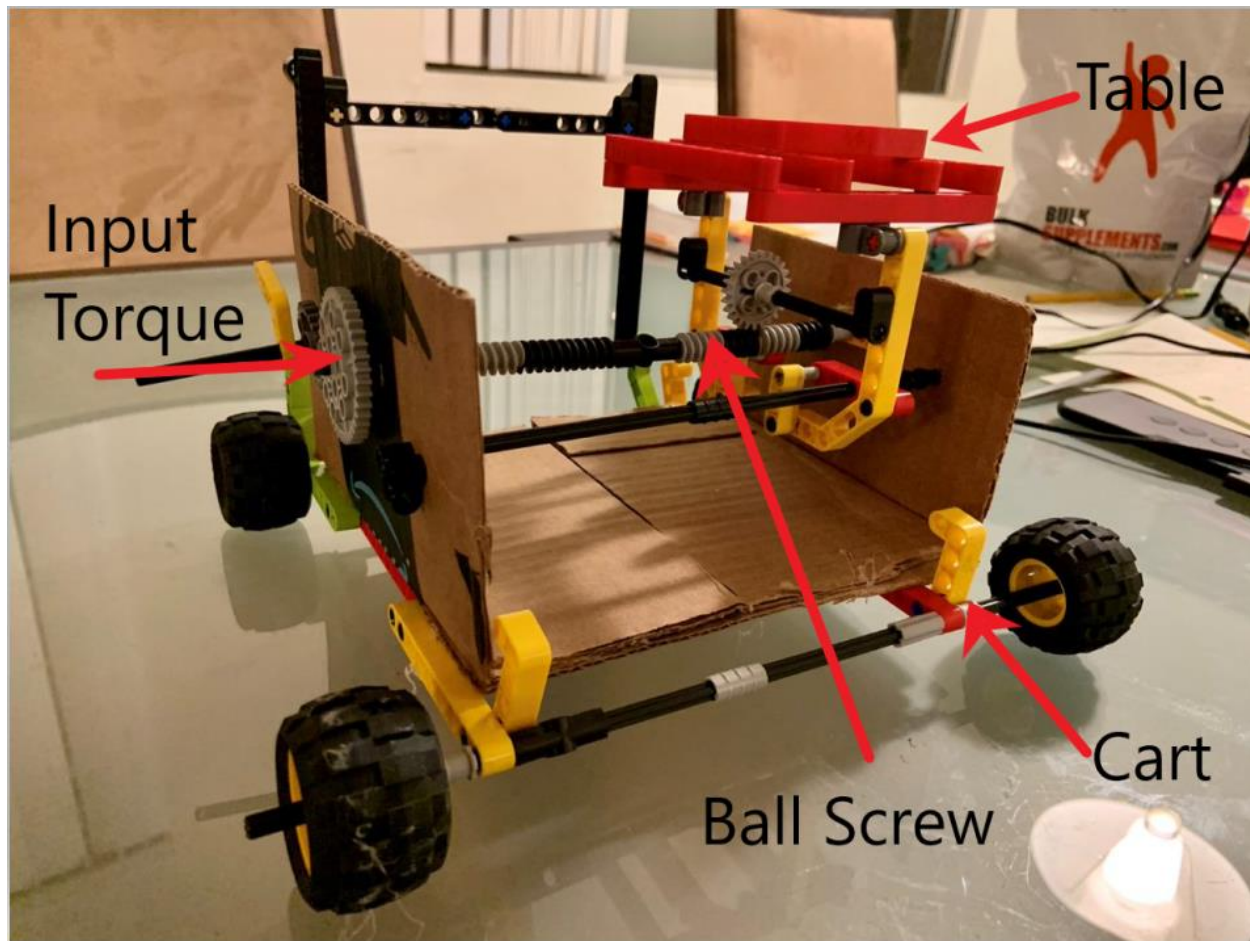


Figure 8: Isometric View of Physical Concept Prototype.

4.4 Preliminary Design Hazards

Before starting detailed design, we identified potential hazards for users of this device using the Design Hazard Checklist, as seen in Appendix A.10. We identified potential risks regarding rotating machinery, pinch points, high forces and accelerations, potential to fall, vibration, and malfunction of the controls system. We have observed existent hazard prevention to include education of users, the use of safety glasses and avoiding contact with machinery during operation, although the implementation of our prototype into a classroom necessitated further considerations. In terms of design, we will need to ensure a critical speed of the ballscrew shaft that is high above the operating point, as well as assuring high strength and stiffness of all components. There is possibility that the controls system that runs the motor will malfunction, which can lead to excessive voltage or motor torque. This can be resolved by adding a Saturation Value to the controls system model, which will effectively put a ceiling on the maximum output voltage that the controls system will demand. Additionally, the use of limit switches and kill switches at extreme points in the table's motion will shut off power to the system before contact is made between two physical components. A hard stop will serve as a final defense in mechanically stopping the table's actuation.

4.5 Anticipated Challenges

While our team is familiar with the overall design process, some areas of the design for this project have special challenges that will take additional research and creativity. Staying within our budget will be challenging, but we expect to be able to remain within our means by comparing prices from multiple

vendors and keeping a Bill of Materials to keep track of all materials and total cost. Electronics design, including communication between the encoder reader and the microcontroller, communication between the microcontroller and the motor, and the implementation of the controls system on the microcontroller is a skill that we will all develop over the course of this project, and will come with hard work and study. Finally, we will need to create a design that is optimized for educational purposes while still being safe and relatable to real world systems. We conducted ideation regarding designing the system to be as educational as possible but will need to implement these designs into our solid model and detailed design.

5. Final Design

Considering the components, functions, and design constraints necessary for our feed drive prototype, we proceeded to compile a final design after accounting for the design choices and concept prototypes made per the previous chapter. Please refer to the following sections for detailed descriptions on part verification, selection, and analysis.

As will be discussed, our final design is composed of a DC motor with rotary encoder, ballscrew, ball and thrust bearings, linear rails, aluminum table and frame, and Nucleo microcontroller with corresponding MATLAB User Interface (Figure 12 for full system assembly).

5.1 Subsystem and Component Verification

The following sections describe the calculations, simulations, and prototyping of system components that were necessary to justify their inclusion in our proposed product. Final design entailed detailed analysis and selection of subsystem components, where the adequacy of our design was determined against the specifications in Table 4 and will be discussed at length below. All mechanical components were designed for a payload mass of 50 lbs (~25 kg). This applied load is far above the table's actual 5kg load carried during experiments, but our intent was to design for misuse, such that an average person leaning on the table would not cause catastrophic system failure. The following sections are organized first by subsystem and then by component.

5.1.1 Drive Train Analysis

Before any components could be selected from off the shelf, we built an analysis tool in Excel to find part specifications that would allow for the fulfillment of system level requirements. We designed parts based on the system specifications in Table 4, with the most driving requirements for our preliminary analysis being table travel distance, load requirement, number of cycles, and speed.

5.1.1.1 Motor

The motor specifications came from a conservative payload mass of 25kg and a maximum speed of 10cm/s. A ballscrew pitch of 5mm was specified based on availability in the market and ability to meet our specified precision of 50 microns, meaning the table must be placed at a physical location within 50 microns of a user's inputted table location in the GUI. We used a conservative estimate for a friction coefficient and motor efficiency to account for variability in the motor's performance and amount of friction between the ballnut and ballscrew. Based on our analysis, we calculated an operating speed of 1200 rpm, steady state drive torque of 0.18 Nm, which equates to a required motor power input of 30.22W based on a work energy equivalence analysis. This analysis can be found in Appendix A.14.

5.1.1.2 Ballscrew

Our ballscrew was designed around four potential failure modes: static loading on the threads, fatigue, buckling, and vibration at natural frequency. For static loading on the threads, we used powerscrew equations from Shigley's Mechanical Engineering Design to compute thread stresses due to friction as well as the normal stress due to bending, which was calculate for the most conservative loading situation when

the table is in the middle of the ballscrew. From there, we computed principal stresses using equations based on Mohr's Circle and utilize the Tresca Failure Theory for a conservative safety factor. After static loading, we investigated fatigue effects on the ballscrew at 99% reliability based on the Mod-Goodman Fatigue Method. During fatigue analysis we accounted for effects due to surface finish, shaft rotation, and reliability. This analysis was completed to ensure a decrease in ultimate strength over the life of the material would not cause yield. We also considered a buckling failure due to compressive loads on the ballscrew, but using Johnson Buckling Criteria we determined that buckling would not be a significant constraint when compared to other failure modes. We found the limiting factor for the ballscrew to be its natural frequency. In the end, the ballscrew we selected yields a critical speed safety factor of 1.559, when driven at a linear speed of 10 cm/s. The safety factors for the other failure modes were all over 100. The spreadsheet used to determine these values can be found in Appendix A.14.

5.1.2 Motion Support

The Motion Support subsystem defines the components that guide and support the table's linear travel. As shown in Figure 9, two subcategories are further identified, namely the "Rail Assembly", composed of linear rails, bearings, and bearing mounts, and the "Ballscrew Support Assembly", composed of thrust and ball bearings.

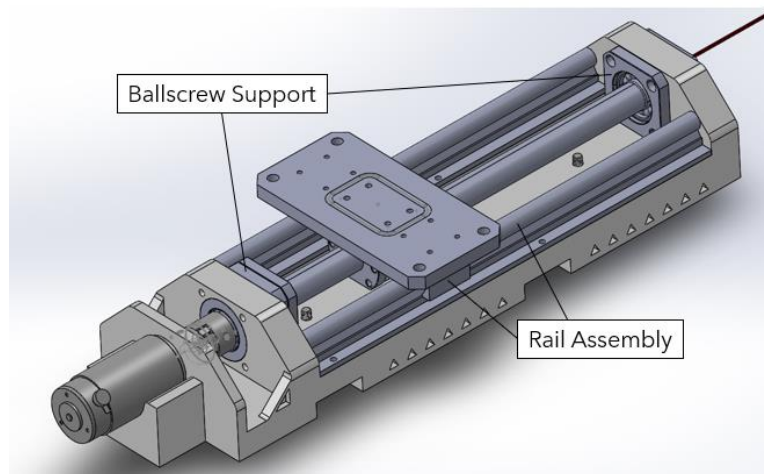


Figure 9: CAD showing both the support components.

5.1.2.1 Rail Assembly

Rail assembly selection was primarily driven by deflection analysis. In practice, we needed to eliminate virtually all bending and binding of the linear rails, which lead our team through the following design methods. Analysis began with the rail sizing excel calculator in Appendix A.14. To determine an adequate shaft diameter, we conservatively modeled the rail as a simply supported beam - a conservative estimate because, in actuality, the rail is fully supported on its undermost face, and fixtured at four hole locations as in Figure 6. A carbon steel material yield strength, taken from railway manufacturer specifications, was applied, along with a shaft length of 30 cm and 50 lb load (safety factor of 4.5x over our applied plate loadings), with which the calculator determined a minimum railway shaft radius of 6 mm.

This minimum radius leads us back to the manufacturer, where we found a rail of proper length at 16mm in diameter (an additional safety factor of 1.3 on top of the conservative estimates already discussed). Finally, we selected rails with a trapezoidal base support, ensuring our parts were fully supported rather than suspended. Our chosen SBR16 rail and bearing pairs were modeled in CAD, then analyzed using the

finite element analysis (FEA) studies in Figure 10. Characteristic features of the SBR16 rails include apt sizing and functional application to the feed drive system and an anti-rust chrome-plated coating.

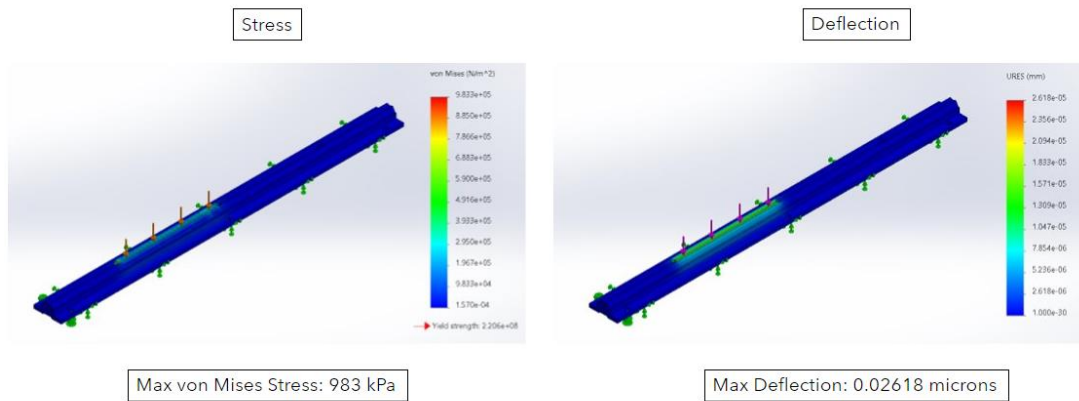


Figure 10: Finite Element Analysis of Linear Rails.

From the railway FEA, we found that under a 50 lb loading, the maximum stress in the rail was 983kPa - under the 220 MPa material yield strength by a factor of 224x - and a maximum of deflection of 0.02618 microns, which is too low impact our 50 micron resolution. Ultimately, our team took consideration to eliminating deflection with each design decision, and the resultant product is well-suited to our specifications.

SBR16UU linear bearing and block components will be inserted concentrically along the rail and will serve as the connections between the table and rail system. The linear bearing will be used to stabilize the table's motion by applying a reaction force at a distance from the ballscrew pivot to oppose ballscrew rotation.

5.1.2.2 Ballscrew Support Assembly

The second motion support subcategory, “Ballscrew Support Assembly”, is comprised of a FF20 ball bearing and FK20 thrust bearing. The ball bearing was designed to support radial loading, while the thrust bearing was designed for axial loading produced by the ballscrew threads. Taking the bearing's projected lifespan of 3000 hours into consideration, we calculated the catalog life of a ball bearing with an applied load of 50lbs on the table. This analysis can be seen in Appendix A.14. Bearings were specified by the manufacturer in tandem with our selected ball screw, thus their selection and purchase happened concurrently. Due to the low axial load on the thrust bearing, and the oversized specifications on the ball bearing from our selected manufacturer compared to our specified catalog life for the ball bearing, we reasoned that the manufacturer's thrust bearing would fit our engineering needs.

5.1.3 Mass Support

The mass support subsystem is comprised of the table and frame structures necessary to accommodate our 5kg load designation and weight of all system components, respectively.

5.1.3.1 Table

Our sponsor indicated that the feed drive must be able to carry a 5 kg payload across a 30 cm travel distance, which prompted our design of a table that satisfies those requirements. In keeping with our goal to create an educational feed drive model, we determined that the payload should be available in incremental plate masses, allowing students to vary how much load they place on the system and observe its subsequent behavior – such as changes in linear speed, system vibrations, and motor output. The table spans the width

of the frame and is fixed atop the four linear bearings and central ballnut, shown in Figure 16. Because the ballscrew and linear rail components have been designed to handle vertical reaction forces of the mass load, as discussed in sections 6.2.1 and 6.2.2, we have allowed for lateral motion at the table attachment holes. The choice of aluminum table material was verified through construction of our structural prototype, with physical confirmation that the table will travel linearly under its 5 kg load without fracture.

5.1.3.2 Frame

To accommodate the placement of our feed drive components, we designed a frame that can be used to mount component geometries in the desired configuration. Clearance holes are provided at the location of the ball bearing, thrust bearing, and rails for a nut and bolt interface, as well as at the motor to interface with the threaded holes on the motor face. Slots under the frame allow for users to screw nuts onto the bolts for the rails and direct wiring safely underneath the system for electrical connection to the limit switches and rotary encoder. An extrude feature was used to bring the rails to the same height as the ballnut bracket for table alignment. In minimizing our system mass, we removed material where we could, with extruded cuts into the body of the frame and a trapezoidal thin-walled shape assigned to the support walls. These features can be observed in the frame schematic in Figure 11 below.

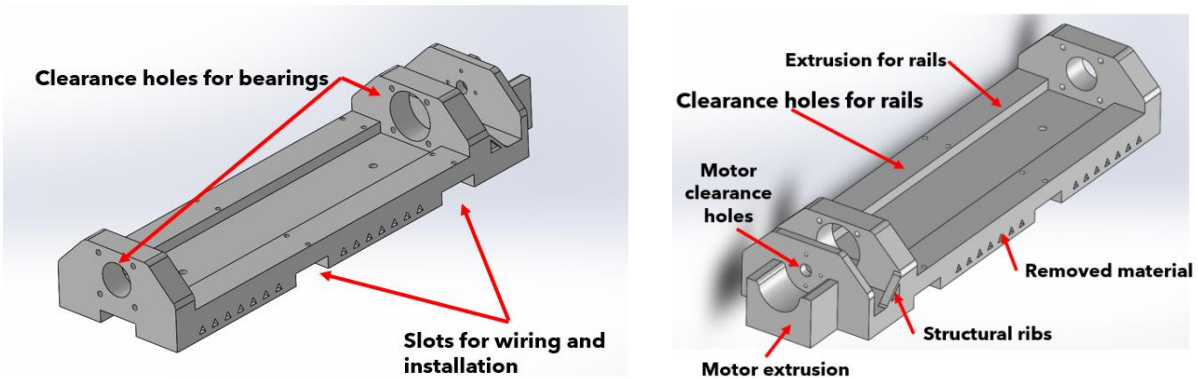


Figure 11:Frame isometric views with key features labeled.

In addition to our kinematic design and mass considerations, we additionally made some structural considerations to minimize frame stress and deflection. We added fillets to sharp corners to minimize the extremity of discontinuities to combat stress concentrations. These sharp corner breaks also serve as an additional safety measure on outer corners. We increased frame area at locations of high force and added structural ribs to provide additional inertia at critical points, such as the support wall for the thrust bearing. Finally, we minimized the distance from support material that high loads would be applied at. By decreasing the lever arm at locations such as the motor mount wall and thrust bearing wall, the applied moment is decreased, which minimizes normal stresses in the material.

We conducted Finite Element Analysis (FEA) in a Solidworks Static Study to verify our design with applied loads to the system. Material properties for PLA, including Elastic Modulus, Yield Strength, Ultimate Tensile Strength, and Poisson's Ratio, were sourced from an MIT Open Source Article^[23]. The assumptions we made for our FEA was for a static, linear elastic, homogenous loading situation. Although the 3D printer prints in layers at different angles, we modeled our analysis as isotropic with conservative material properties. We applied conservative vertical loads of 120N at both bearings' locations and a 100N axial load distributed over the thrust bearing wall. We assumed a fixed support at the table, and we used a reasonably fine mesh to obtain a precise analysis with reasonable computational time. We observed a maximum Von Mises Stress of 383kPa, well below the literature yield strength of 70 MPa^[23]. We

additionally observed a maximum deflection of 16.1 microns, which is within reason for a system precision of 50 microns.

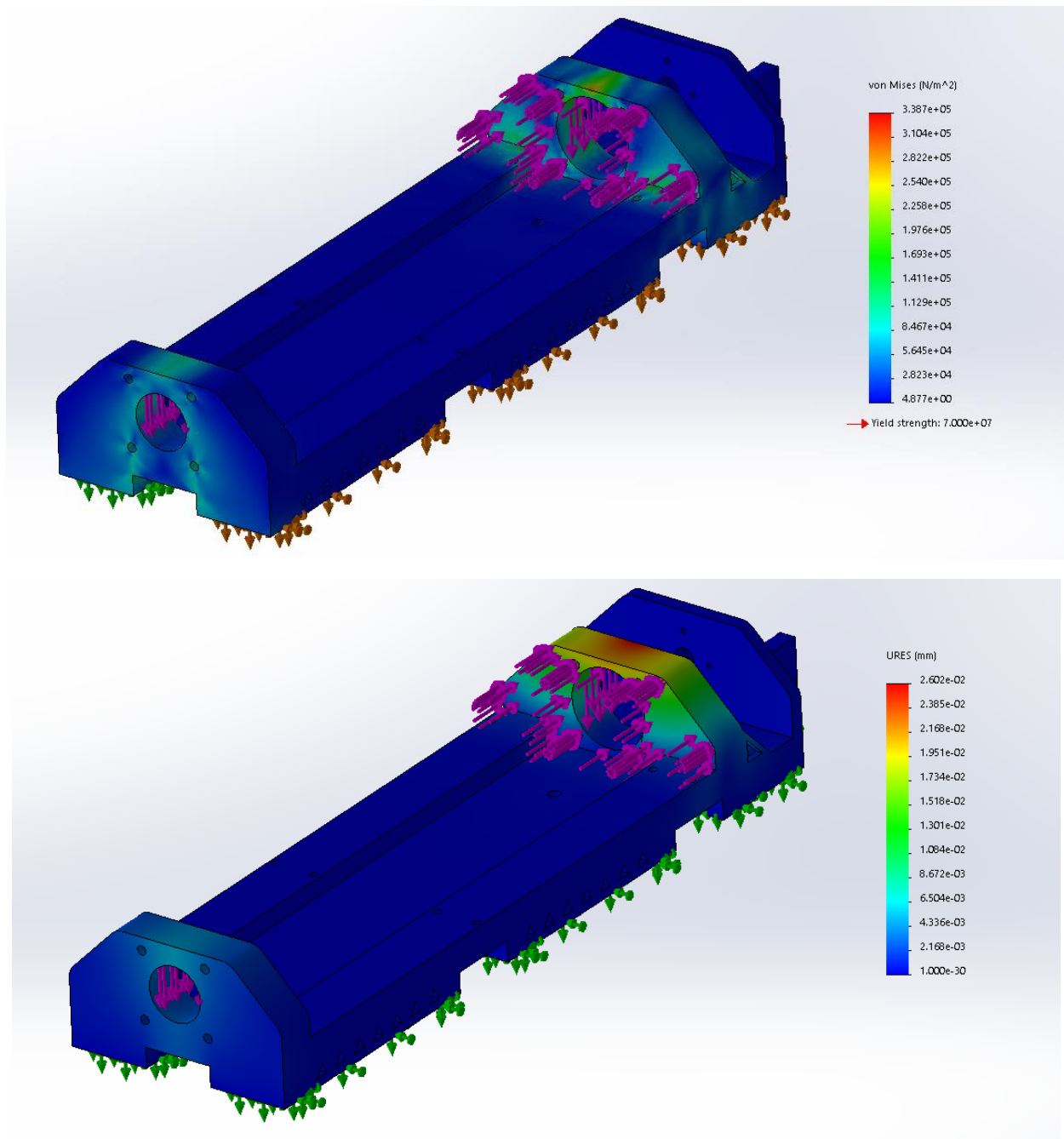


Figure 12: Solidworks FEA performed on the frame to ensure reasonable strength and stiffness. A maximum Von Mises stress of 383kPa and a maximum deflection of 16.1 microns is observed.

Initially, we considered the frame to be printed as a single part out of Polyactic Acid (PLA), but found that manufacturing complexities and questions on prolonged structural integrity led us to design the frame out ¼” and ½” 6061 Aluminum stock, where its design was greatly simplified to be three vertical walls fixed

to a bottommost base plate. This conversion offered more effective manufacturing methods and was afforded by selecting a 12V DC servomotor with rotary encoder attached.

5.1.4 Controls

To assist with the development of our controls system and user interface, we have recruited Samuel Wong, a Computer Engineering student, as an interim member of our team for the remainder of the project. The following section will cover our current progress on our controls, and briefly touch on our plan for further development.

5.1.4.1 Microcontroller and GUI

A Nucleo L476RG microcontroller will be used to interface with the encoder, motor, and the PC that controls it. The PC will communicate with the microcontroller via a serial port, which will be facilitated with a USB cable. A GUI will be created for ease of use, and a simple-to-implement GUI library will be used for possible future changes to the program. The GUI is currently setup to take the position and proportional gain as the inputs to the control system. The prototype GUI model can be seen in Figure 13. below. It currently reads out positional data on a moving graph using the Matplotlib python library. Improving the graphical and functional appeal of the GUI, as well as implementation of the controls system will be a major focus moving forward into spring quarter.

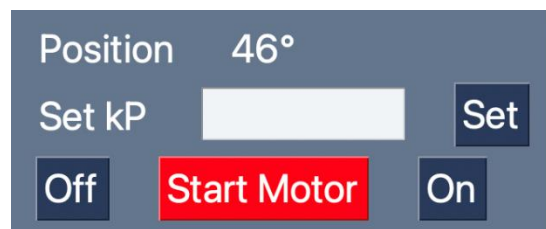


Figure 13: The current dummy GUI used for prototyping.

5.1.4.2 Rotary Encoder

We decided to switch our design direction from using a linear encoder to using a rotary encoder due to lower complexity and cost. We have selected a rotary encoder with 14-bit resolution that fits our resolution requirements and uses a SPI output, which is compatible with our microcontroller. The rotary encoder will mount onto the encoder adapter, which will be coupled with the same frame clearance holes that the ball bearing housing is concentric with. The rotary encoder spindle will connect to the ballscrew shaft end via an encoder coupling. Both the encoder coupling and encoder adapter were designed by our team and included in the drawing package to be 3D printed during the manufacturing process. A schematic showing the components for connection between the ballscrew shaft, and the encoder can be seen in Figure 14 below.

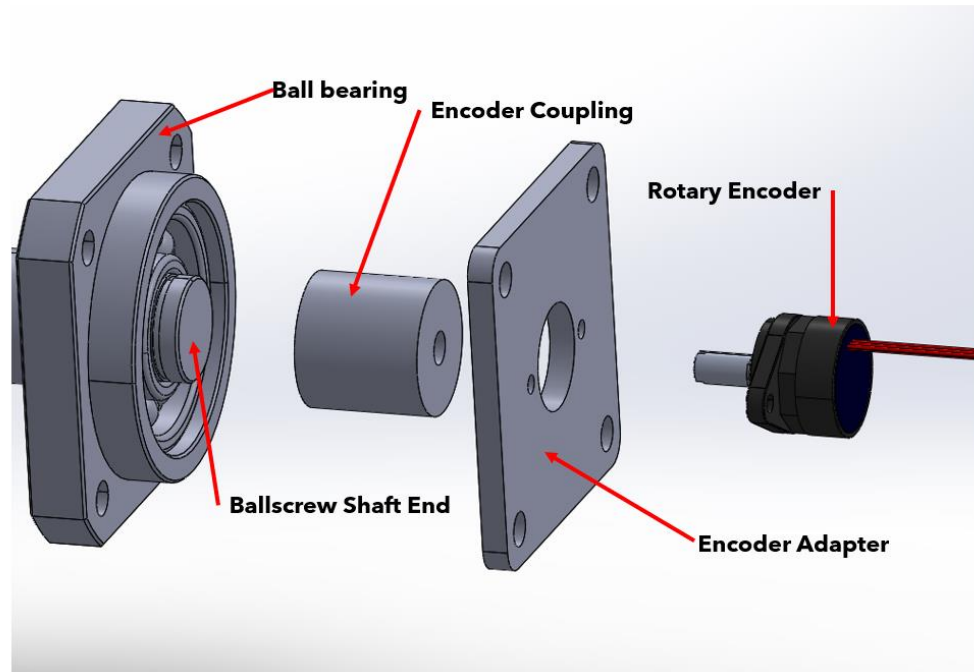


Figure 14: Encoder mounting scheme from rotary encoder to ballscrew shaft.

5.2 Final Design Selection

After developing component specifications based on concepts rooted in engineering mechanics, we moved on to detailed component selection and assembly into a final CAD model. All pieces in the assembly were designed with clearance from other components, and the frame design served as the foundation in securing parts. All components are to be purchased from external vendors except for the frame, table, encoder adapter, and encoder coupling which will be 3D printed. With no threads designed into the frame, nuts and bolts will be used to assemble the bearings and rails onto the structure. The rails were cut down in CAD to match the travel length of the ballscrew, which will need to occur with a circular saw during manufacturing. For motion control, the rotary encoders to monitor table position will be mounted on the ballscrew shaft adjacent to the ball bearing, with limit switches placed at the extreme ends of the table's motion. A full assembly schematic of our CDR CAD can be seen in Figure 15 below.

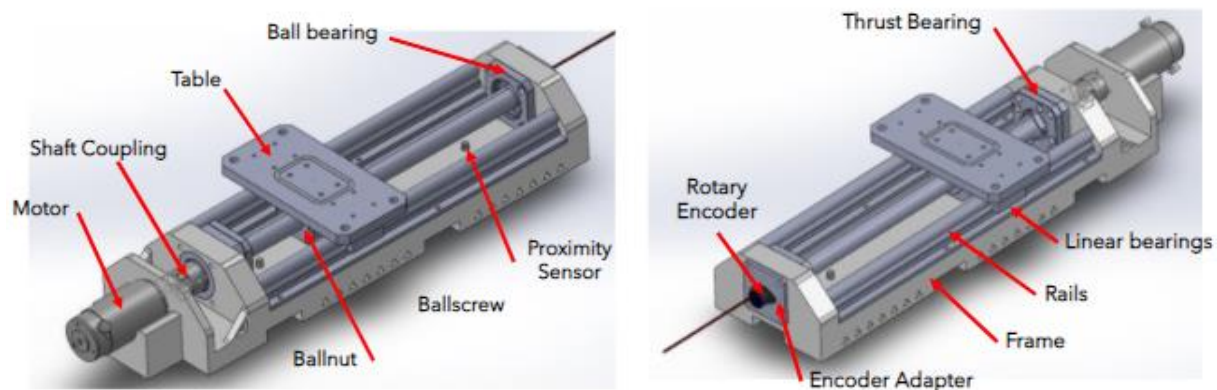


Figure 15: Full Assembly with final parts.

5.3 Safety, Maintenance, and Repair Considerations

We are committed to designing a system that is both effective and safe for the user. Because of this, our team created a Failure Modes and Effects Analysis, which is in Appendix A.12. This analysis investigates the potential failure modes while describing the potential effects, causes, preventive activities, and current detection activities of these failure modes. The team has included a factor of safety for all our components and has considered worst-case scenarios.

Most of the system safety precautions come from the drive train and motion support components failing due to the transfer of motion from the electric motor which could cause deflection or fatigue. The team has run some analysis on these components presented in Appendix A.14.

Other protective measures include an exposed kill switch on the microcontroller to prevent extraneous motion of the masses and effective cable management to prevent any wiring catastrophes. Our goal is to make sure it is difficult for the user to do something dangerous with the device. The team developed a step-by-step assembly plan that will describe to the user how to replace parts if necessary, located in the Manufacturing Plan in Section 6.

5.4 Cost Analysis Summary

After researching vendors and compiling their prices into an indented Bill of Materials in Appendix A.11, the total cost of the system came to \$761.82. However, there is a good chance we will be over budget due to the hardware we need to obtain for the controls, and we will need a power supply. We are planning on contacting Dr. Birdsong to request some funding through the Controls Lab budget. The bulk of the system's costs will come from the electric motor (\$189.61) which is justified based on the requirements that it satisfies.

Cost for our drivetrain and motion support function were able to be minimized by one vendor (Automation4Less) that provided a great deal on components like the ball-screw and bearings. Also, since our larger components like the frame and table will be 3D-printed this minimized the manufacturing cost and only cost-effective spools of PLA needed to be purchased.

The team is confident they can stay under budget as the project continues even though they still need to purchase their control system components. As they assemble to structural prototype, they will have wiggle room when it comes to purchase extra or replace parts. In addition, as they run their design test, they see what needs to be changed. A summary of major component cost is shown below.

Table 7:Summary of Costs

Component	Approximate Cost
Electric Motor	\$189.61
Ball Screw	\$100.00
Ball nut Bracket	\$31.50
Rail and Linear Bearing	\$30.00
Ball Bearing	\$36.00
Thrust Bearing	\$95.40
Microcontroller	\$14.60
Fasteners	\$20.00

5.5 Structural Prototype

For our structural prototype, our team decided to order the final parts for the drive train and 3D print the table to observe the actuation of the table with our own eyes. We wanted to ensure that for a given turn of the input shaft, the mass supported on the table would actuate predictably. After purchasing the necessary components, we started by securing the ballnut bracket to the ballnut with the use of 6 M6x1 screws. The ballnut was installed onto the ballscrew and then each of the bearings were press fit onto the ballscrew with the use of a rubber mallet and WD-40 lubrication. Finally, we 3D print the envelope of the table using a Creality Ender 3 Printer and mounted this table on the ballnut bracket using 4 M6x1 screws. An overhead picture of our structural prototype can be observed in Figure 16 below.

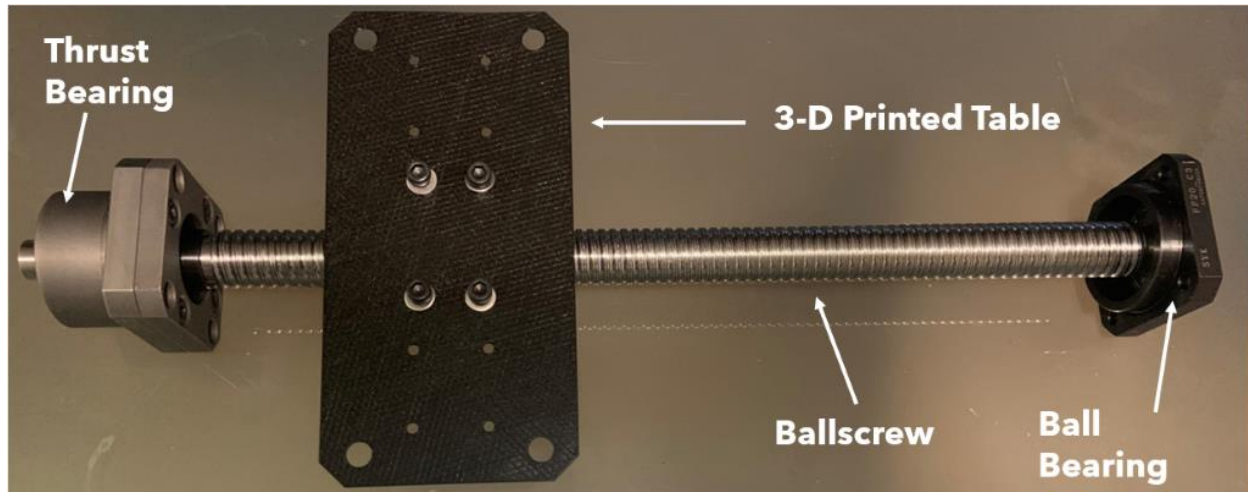


Figure 16: Schematic of the structural prototype

The opportunity to put our hands-on hardware brought a few immediate lessons to our attention. The vendor from which we sourced our powertrain components did not provide part mass, so we got to experience with our own hands the significant mass of the ballscrew and bearings. We experienced difficulty 3D printing large masses from PLA using our personal 3D printer, which brought us to consider a custom print from a manufacturer. Ultimately, after receiving quotes outside the budget range, our system was redesigned to be manufactured from $\frac{1}{4}$ " 6061 aluminum stock.

5.6 Post CDR Design Changes

Due to the low availability of 3D printers that would support the print for a piece as large as our frame, we redesigned our frame from Delrin, a machinable plastic. After consulting with Eric Pulse, we received the recommendation that the deformation of Delrin during machining will be too significant for the accuracy that our system requires. We learned about the devastating deflection that heat of machining can have on precision components, so we redesigned our frame again from aluminum. We decided that we would cut stock metal and machine the pieces ourselves, dividing the frame into motor face, base plate, ball bearing face, thrust bearing face, table, spacers, and legs. Due to the acceptable strength and cheap cost, we decided on an aluminum thickness of 0.25in. The only exceptions were the spacers and thrust bearing face being machined from 0.5in. stock for dimensional constraints and minimal deflection respectively. We decided to connect all components with threaded fasteners for both strength of connection and ease of disassembly when needed, with the use of 90-degree angle brackets to attach the outer walls to sides of the base plate. All fasteners are to be secured with nuts. We implemented two counterbored legs in our design to elevate the base plate to allow for nuts to be fastened on the bottom of the structure. A picture of our final CAD model can be seen in Figure 17.

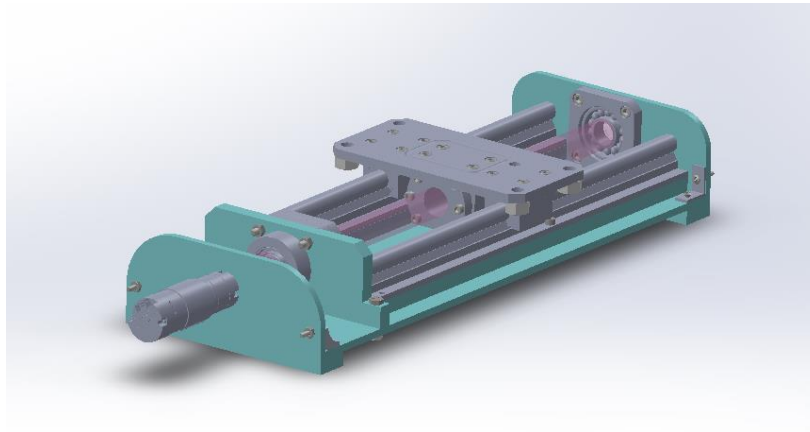


Figure 17: Final CAD assembly, with the implemented aluminum structure replacing the 3D Printed frame. The structure consists of a base plate, three walls, two spacers, and two legs.

In addition to this structural change, we found an economical face-mount motor with built in encoders with speed and torque specifications within our range from Servocity. While a motor with a built-in encoder does not as clearly visually illustrate the control loop upon which our control system is based, it is more like systems encountered in real life and provides for a more compact total design. This addition allowed us to neglect our original encoder mounting design and focus simply on a face mounted single unit motor and encoder.

Additionally, while we originally planned for optical limit switches in our final design, we switched to two purely mechanical roller switches operating as normally closed, such that contact with the switch opens the circuit and causes the pin reading from the switches to read contact. This change was driven by the special constraints on the sides of the base plate, the lower cost, and less effort, as the original optical switches that we sourced were designed for much smaller applications.

6. Implementation

Due to the machining precision required and the market readiness of the required parts, we sourced and purchased most of the feed drive's components from 3rd party vendors. We manufactured the aluminum frame that we designed as discussed in Section 5.6 based on the produced drawings that can be found in the drawing package in Appendix A.12. We made great use of the Cal Poly Machine Shops for manufacturing, including The Hanger and Mustang 60. Final assembly and implementation of coding took place in San Luis Obispo.

6.1 Part Procurement

As we approached Verification Prototype Sign Off, we still needed to acquire a few components to give a clear representation of our system. These parts included the Motor Driver breakout board purchased from ST Microelectronics, the limit switches, purchased from Amazon, shaft coupling purchased from McMaster Carr, the lid purchased from Target, and the aluminum sheets we purchased from Grainger. The remaining electrical parts included the motor which we purchased from ServoCity, the Nucleo microcontroller from ST Microelectronics, and the 24V DC power supply from Amazon. The parts added to our inventory before the CDR included the ballscrew, the two supporting bearings, the ballnut, and the ballnut bracket which we ordered in a package deal from online supplier Automation4Less. Similarly, we purchased the rails and linear bearing, as well as the PLA for the printed components, from Amazon.

6.2 Final Budget Status

After the completion of the verification prototype, our final cost amounted to \$758.54, including tax costs and shipping fees. This left us with \$32 remaining from our \$800 budget to be put towards any future modifications by our sponsor.

Table 8: Summarized Cost of Componentry

Items Purchased	Vendor	Subtotal	Shipping/handling/tax
PLA Filament	Amazon	\$22.99	\$1.78
Ball Screw, Ballnut, and Bearings	Automation4Less	\$262.90	\$20.77
Rails and linear bearings assembly	Vevor	\$29.99	\$-
NUCLEO-L476RG (Microcontroller)	ST Microelectronics	\$22.59	\$4.35
Limit Switches and shaft couplings	McMaster-Carr	\$12.68	\$-
Aluminum Sheets	Grainger	\$194.82	\$44.91
Motor Driver	ST Microelectronics	\$12.88	\$5.00
Power Supply	Amazon	\$13.99	\$1.22
Roller Switches	Amazon	\$6.99	\$0.61
Motor Driver	ST	\$12.88	\$-
56 Fasteners and Corner Brackets	Ace Hardware	\$33.31	\$2.91
Female Bullet Adaptors	ServoCity	\$3.99	\$-
Motor	ServoCity	\$39.99	\$6.99

6.3 Manufacturing/Assembly Process

The following section details the assembly process of our system. It will also touch on the keyframe manufacturing steps that we performed in the Mustang 60 and Hangar shops.

Drive Train

Motor (111000)

Purchased

Assembly

1. Placed the motor securely against the motor mounting face and aligned the four counterbored M4 Clearance holes with the M4 tapped holes on the motor's face. We secured this connection with four M4 fasteners.

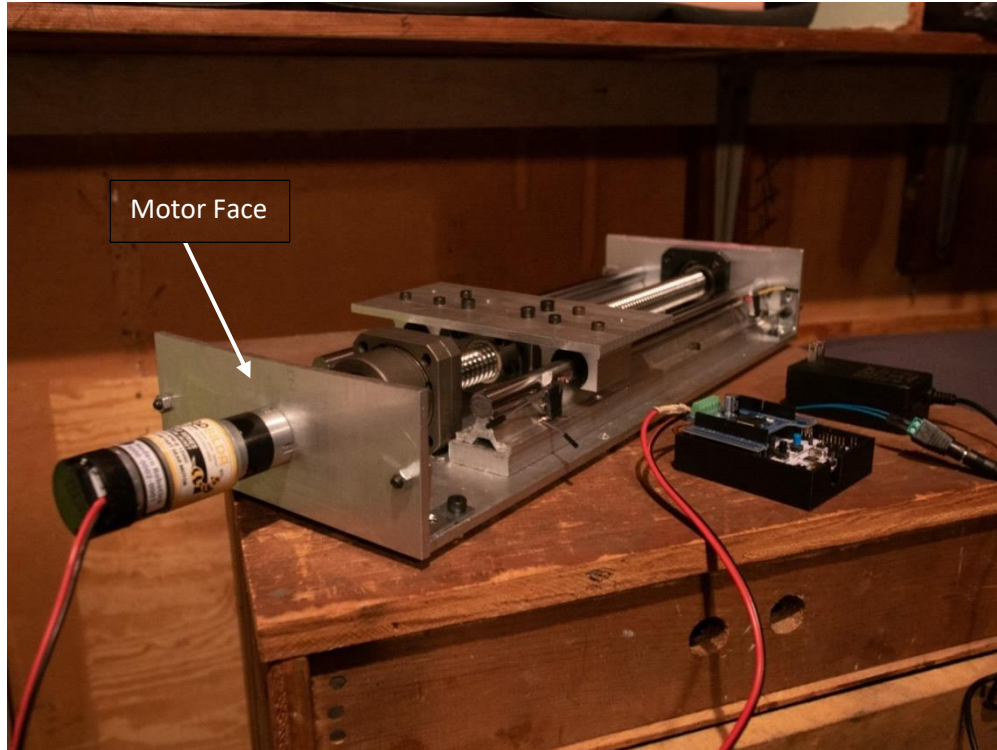


Figure 18: Isometric view of the assembly, with the motor driving the ballscrew via flexible shaft coupling. Channel A leads from the motor driver the motor, powered by a 24V wall outlet source and controlled by a Nucleo Microcontroller.

2. Placed the motor shaft inside the small diameter side of the flexible shaft coupling.
3. Secured flexible shaft coupling as described below in “Flexible Shaft Coupling.”
4. Connected motor leads to the bullet lead to motor driver adapter, which is connected to the Motor A output pins on the Motor Driver board. Secured this screw terminal connection with a flat head screwdriver.

Ballnut (114000)

Purchased

Assembly

1. Screwed the ball nut onto the ballscrew by spinning the ball nut onto the ballscrew. We completed this by flipping the ball nut onto the ballscrew top and screwing the ball nut onto the screw until the arbor disconnects from the ball nut.

Ball Screw (113000)

Purchased

Assembly

1. Press-fit free machined end of ball screw into the ball bearing.
2. Used Circlip/Snap ring to secure screw to the ball bearing.
3. Press fit the fixed end (17mm) side of the ballscrew through the thrust bearing (Motor Side).
4. Positioned machined end in the flexible shaft coupling and securing it with set screws. Followed instructions below under “Flexible Shaft Coupling” before proceeding.

Flexible Shaft Coupling (116000)

Purchased

Assembly

1. Ensured the flexible shaft coupling rested in the mounting area and that the motor shaft and ballscrew end contacted the inner surface of their respective coupling hubs.
2. Ensured that the two coupling hubs firmly gripped the spider.
3. Tightened the set screws on both coupling hubs to fix each shaft to the hub by through the friction of the set screw. Ensure that both hubs still contact the spider.

Ballnut Bracket (114000)

Purchased

Assembly

1. Fastened the ballnut bracket to the ballnut by aligning the threaded holes in the hexagonal pattern on the bracket's side with the flange on the ballnut and clamping the two components together with 6 M8x0.8 screws.

Motion Support

Rails and Linear Bearings (121000)

Purchased

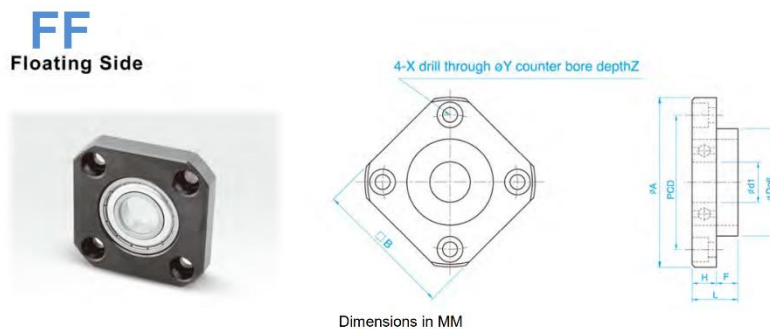
Assembly

1. Fastened rails to the bottom of the frame and linear bearings will run the rails and mounted to the mass support table.

Ball Bearing (122000)

Purchased

Assembly



Figure_ . Drawing for the ball screw ball bearing.



Figure 19:Expanded view of Assembly.

1. Positioned the bearing on the opposite side of the ball screw to allow for ball screw rotation.

2. Ran the ball screw through the FF mounted body then the sealed ball bearing and lastly secured by the snap ring.
3. Mounted to the frame using four M6x1mm thread, 8mm long screws.



Figure 20: Ball bearing mounting configuration on the ball bearing mounting face. The ball bearing protrudes through the ball bearing mounting face and is secured with four M6 nuts and bolts.

Thrust Bearing (123000)

**Purchased
Assembly**

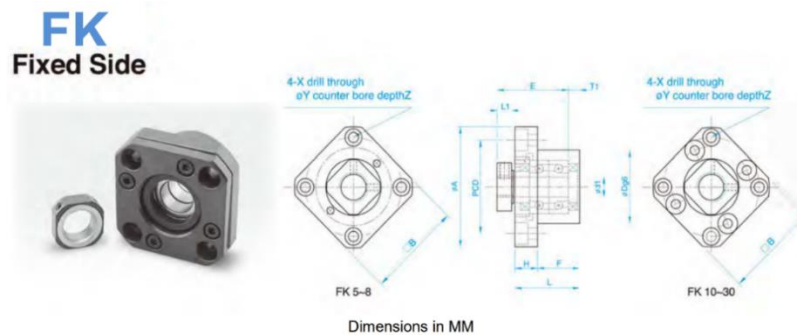


Figure 21: Drawing for thrust bearing.

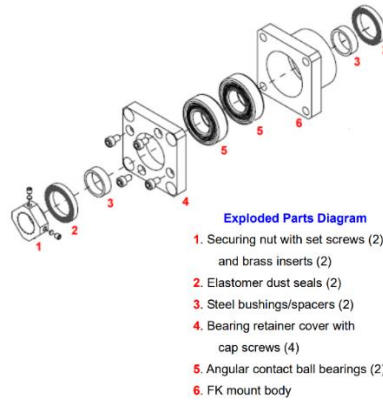


Figure 22: Expanded view for assembly.

1. Positioned the bearing on the motor side of the ball screw.
2. As seen in the diagram above we ran the ball screw through those parts and secured it by the securing nut with set screws.
3. Mounted to the frame using the same M6x1mm thread, 8mm long screws.

Mass Support

Table (132000)

Vertical Bandsaw

1. Cut the table from stock Al-6061 0.25" stock to length. Sanded to smooth the edges to ensure flatness.

Drill Press

2. Drilled four M6 clearance holes as measured for alignment with the ballnut bracket and eight M5 clearance holes for alignment with the linear rails on both sides of the table.

Assembly

3. Mounted the table onto the ballnut bracket by aligning the four clearance holes in the middle of the table with the 4 M6x0.8 tapped holes on the ballnut bracket. Secured the connection with fasteners.
4. Secured the table to the linear bearings with M5 screws.

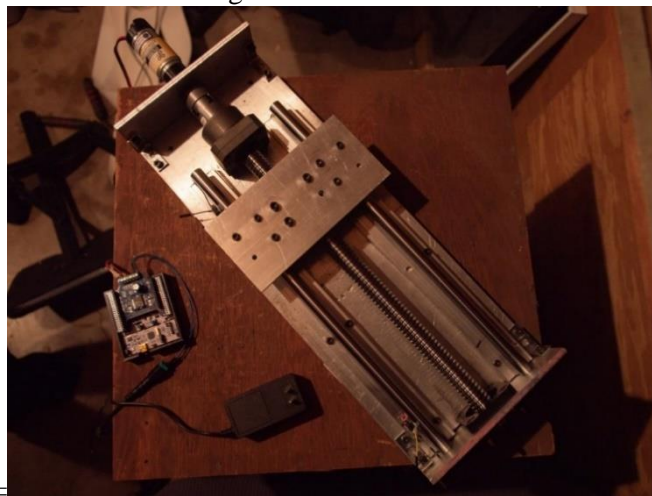


Figure 23: Top view of the system depicting the length of the ballscrew, with the ball nut connected to the table and driving on the linear rails.

Structure

Frame-Main Manufacturing Process (142000)

Assembly

1. Aligned each of the three walls (datum face, motor face, ball-bearing face) with their respective clearance holes.
2. Fastened the pieces together with the two clearance holes at the side of each component. Each screw meshed with a nut to secure the pieces of the frame together.

Note: The frame underwent redesign over Winter Quarter, with our team deciding to manufacture the previously singularly printed frame as a four-piece aluminum structure consisting of a base plate, datum face, motor face, and ball bearing face. We joined these pieces together using L-brackets.



Figure 24: Cut out of all frame pieces.

Base Plate (Manufacturing)

1. Cut Al-6061 0.25" stock to length (plate width matches the 8" width of the stock). Used a 2-flute 3/8" end mill on the shop mills in Hangar.
2. Used appropriately sized drill bits on the mill's drill chuck to cut the L-bracket, linear rail, and datum face clearance holes.

Motor Face (Manufacturing)

1. Used vertical saw in the Hangar to cut Al-6061 0.25" stock to length dimensions of 2.93x8 in.
2. Used a 2-flute 3/8" end mill to trim and clean up edges in Mustang 60.



Figure 25: Flattening and squaring the edges of the motor face on the Hangar mill.

3. Used a combination square to scribe hole locations.
 - a. Hole Designations
 - i. Four 0.45" in thru-holes.
 - ii. One .090" thru-hole is all cut by the drill press.
4. Used appropriately sized drill bits on the mill's drill chuck to cut the clearance holes of the motor face in the Hangar.
 - a. Hole Designations:
 - i. One M9 clearance hole for joining of the motor spindle and flexible shaft coupling.
 - ii. Four M5.5 clearance holes for mounting of the motor onto the face.

Datum Face (Manufacturing)

1. In the Hangar cut Al-6061 0.25" stock to length (face width matches the 8" width of the stock). Used 2-flute 3/8" end mill on the shop mills.
2. Cut legs on either end of datum face as per the drawing. These legs allowed the datum face to be pin-located atop the base plate.
3. Measured out holes using a combination square, digital calipers, and scribe to fit 50-micron tolerance. Punched drive indents to make hole locations clear when using the mill. (Hangar).



Figure 26: Measuring and marking hole locations on the base plate and datum face using a combination square, digital calipers, and scribe.

4. Started the center hole of the datum face using a 2-inch diameter hole saw on the mill in Mustang 60.



Figure 27: Cutting a starting center hole of the datum face using a hole saw.

5. In increments of 0.020", reamed out the rest of the center hole using a bore bar on the Mustang 60 Mill.



Figure 28: Using a bore bar on the Mustang 60 mill to cut the center hole of our “Datum Face”.

6. Lost the set screw of the bore bar in the only crevice of the Mustang 60 Mill. Hired a shop tech to find the screw as it was the only size that did not exist in the shop’s extra hardware bins.



Figure 29: Getting help from a shop tech to find a lost set screw in the Mustang 60 mill. He found it!

7. Used appropriately sized drill bits on the mill’s drill chuck to cut the clearance holes of the datum face in Mustang 60.
 - a. Hole Designations:
 - i. Two ¼-20 clearance holes for locating the face atop the base plate.
 - ii. Four M6 clearance holes for mounting of the thrust bearing onto the face.

Ball Bearing Face (Manufacturing)

1. Cut Al-6061 0.25” stock to length (face width matches the 8” width of the stock). Used a 2-flute 3/8” end mill on the shop mills. We performed these steps in the Hangar.

2. Cut four M6 clearance holes for mating with the ball bearing housing clearance holes using a drill press in Mustang 60.
3. Used a 58mm hole saw bit on the drill press to drill a clearance hole for the ballscrew at the specified position in Mustang 60.

Assembly

1. Used L-brackets with M6 screws to secure the motor and ball bearing faces to the base plate.
2. Drove a 1/4-20 pin through the vertical holes on either side of the datum face to secure it to the top of the base plate.

Lid (142000)

Purchased

Assembly

1. Place the lid on the frame.

Controls

Microcontroller (151000)

Purchased

Assembly

1. Mounted Nucleo microcontroller to the side of the housing in the MCU bracket
2. Connected wires based on wiring diagram from MCU to motor via jumper cables, MCU to rotary encoders via jumper cables, and MCU to computer via USB connection.

Microcontroller Housing

Manufactured

1. Redesigned a Nucleo part file that was posted on a shared community design drive.
2. Made changes to fit our current setup.
3. Created an stl. File and 3D printed the enclosure.

Assembly

1. Placed Nucleo three-hole cutouts on the enclosure mounting pillars.

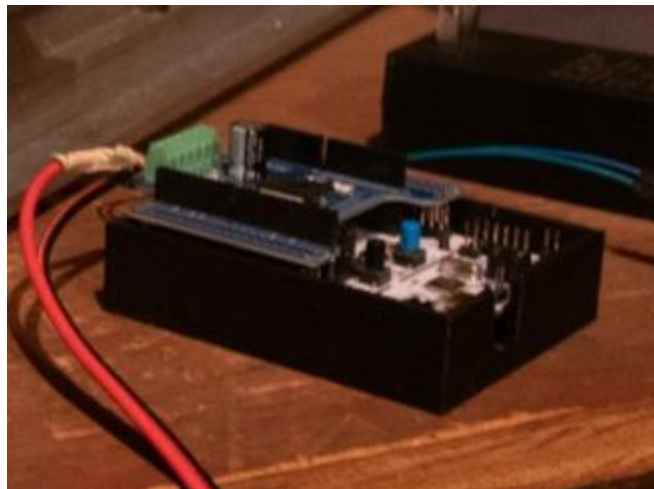


Figure 30: Enclosure with Nucleo mounted.

Rotary Encoder (152000)

Purchased

Assembly

1. Integrated into motor upon purchase.
2. Connected jumper cables of power, ground, Channel A, and Channel B to the microcontroller based on the wiring diagram.

Limit Switches (153000)

Purchased

Assembly

1. Cut two female jumper cables to 2-inch length for each limit switch
2. Strip half of the cable with a wire stripper to reveal the inner copper. For one of the two female connectors, we wrapped the wire three times around the “C” terminal on the switch, and for the other wire, we completed this wrap around the “NC” Terminal (Normally Closed).
3. Soldered each of these two wires to the switch to ensure structural integrity.

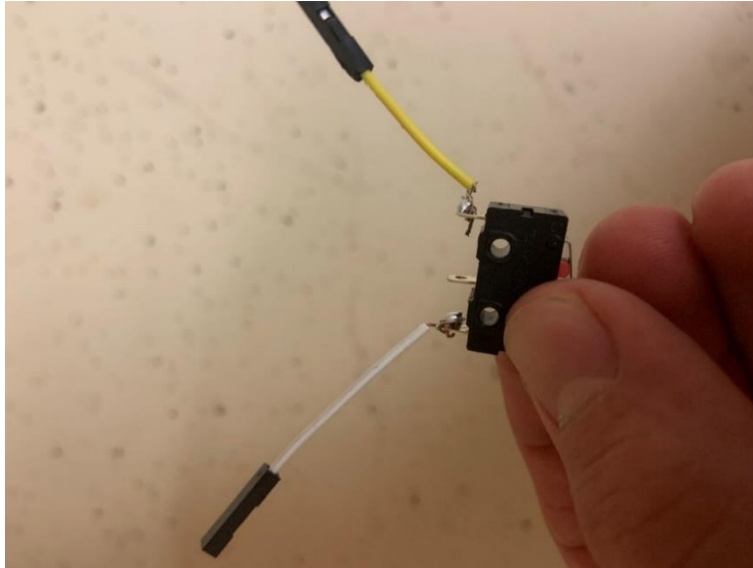


Figure 31:A soldered connection between cut wires and switch terminals. We soldered one wire to the “Common” terminal and one wire to the “Normal Closed” terminal.

4. Attached these switches at the extreme position of the table’s motion using Command Strips.
5. Connected wires from optical sensors to the MCU based on the wiring diagram.



Figure 32: Limit switch mounting configuration. We mounted the limit switches at the extreme points of the table's motion on the outside of the rail and are fixed with command strips. The switches are normally closed but open the circuit upon contact with the linear bearing.

Much of our work would not have been possible without the guidance of the Cal Poly Machine Shop Technicians. Also, we are thankful to Eric Pulse and Professor Schuster for organizing the opportunity to obtain our yellow tags late in this project.

6.4 Microcontroller Programming

To allow students to test their controller gains and perform system characterization in the Controls Lab, we developed the software that ran the hardware. This involved the development of a main file, `main.py`, which controlled the timing, frontend communication, and data collection for our system. This main file creates 4 objects to drive our hardware, cut from the other classes that we wrote: `controller.py`, which directs the control system and drives all the hardware as timed by `main.py`, `EncoderLab6.py`, which interacts with the encoder connected to the motor spindle, `MotorDriver.py`, which drives the motor through Pulse Width Modulation (PWM), and `LimitDriver.py`, which interacts with the two limit switches. The documented code for the five backend files is attached in Appendix A21.

The main file `main.py` runs upon startup of the microcontroller, initializing serial communication and awaiting a comma separated list sent by the frontend, containing the three controller gains K_p , K_i , and K_d , as well as three reference positions, whether the system is to be run in closed loop or open loop, and the open loop duty cycle for if the system is to be run in open loop. After this information is obtained, `main.py` initializes all relevant objects and prompts the controller to initiate startup, which causes the ballnut to move in open loop in the negative direction until a limit switch is engaged, at which point the encoder is zeroed and the ballnut moves 1cm back in the positive direction.

While running in the closed loop configuration, the controller attempts to reach steady state at each reference position by spinning the motor and moving the ballscrew based on the voltage specified by the implemented PID control system. Runs of the control loop are initiated at regular intervals specified in `main.py`. The current interval implemented in `main.py` is 30ms. When steady state is reached, the reference position in the control system will be set to the next input reference position. We have defined steady state for our system as within 0.5mm of the reference position and with an applied voltage of less than 16% of the maximum 12V input. This voltage constraint accounts for the fact that the system may overshoot. While the motion system resolution is within the specified 50 microns, reaching steady state takes excessively long to reach due to static friction in the system. Upon each run of the controller, time data, linear position in centimeters, and linear velocity in centimeters per second are sent back to the front end for real time

plotting via serial communication using the built in Pyboard UART module. After the table has reached all three positions at steady state, the word 'DION' is written over the serial port to communicate that data collection has terminated.

While running the system in the open loop configuration, the requested PWM duty cycle is applied to the motor until a table travel distance of 19 cm is observed by the encoders. This open loop configurations allows for system characterization, such as an experimental determination of motor steady state gain and system time constant, as opposed to the controller design aspect presented by the closed loop configuration. The entirety of main.py is constructed inside of a While(True) loop, which allows a successful run of the system to be followed by another run from the frontend, without requiring that the system be powered down.

The Motor Driver file, MotorDriver.py was heavily implemented from a similar file that we coded during ME 305. The Motor Driver uses three pins to run, with two used for PWM and the nSLEEP_Pin used to enable and disable the motor. The motor is enabled by setting nSLEEP_Pin high and disabled by setting nSLEEP_Pin low. The "set_duty()" method is used to set the duty cycle of the motor, with a saturation limit placed in the class such that only a 50% duty cycle can be applied. This saturation limit exists because we are running a 12V motor from a 24V source.

We also implemented our Encoder Driver based off the encoder driver that we constructed for ME 305. The update() method updates the current encoder position and corrects for counter overflow by correcting the observed delta. If the controller calls for the encoder to zero, such as during startup, this occurs in the update method. All other methods return linear and rotational position and velocity, with the units that are convenient to the user. Rotational position in revolutions is computed from the encoder ticks times the encoder CPR, tabulated as 25.9 for the motor output shaft, and multiplied by 4 since our encoder is quadrature, which means that one encoder count is the equivalent of 4 ticks. Linear position can then be derived by the pitch of the ballscrew, for after one rotation of the screw, the ballnut actuates the length of the pitch, found to be 5mm. Finally, since we prescribe the interval at which the controller updates the encoder, the linear and rotational speeds are simply the change in position divided by change in time.

The simple limit switch class, stored in LimitDriver.py uses only an input pin with a connected pullup resistor. The read() method returns "True" if the pin reads high, which signifies an open circuit, and returns "False" if the pin reads low, which signifies that the circuit is closed.

When the controller object is created by main.py, all gains, the initial reference position, and all hardware driver objects are passed to the controller for full control of the system. While main.py controls the timing and serial communication, controller.py acts as the full control system, taking data from the encoder and dictating the voltage to the motor. After startup, the run() method is the main method used to control the system. This method runs a single iteration of a closed loop controller. Encoders update, the controller denotes the voltage to output to the motor, and the duty cycle is set as a percentage to the motor. If a limit switch is engaged due to overshoot of the system, the controller disables the motor. Data is then returned to main.py as a tuple of the form:

[position[cm], velocity[cm/s], angular velocity[rpm], motor duty cycle [%]]

The control scheme used by the controller that acts on the error is a PID controller. Additionally, students can implement P, PI, or PD control by setting non-relevant controller gains to 0. The P control acts as a spring, pulling the table toward the reference position proportional to the error. The integral control is used to minimize steady state error, for the longer that the error is non-zero, the harder the motor will push to correct the error in position. This control was implemented as a discretized integral, with e_sum storing the

sum of the products of all previous errors and time interval to yield the area under an error-time curve. The gain K_i acts on this e_sum , and e_sum is reset when a new reference position is requested. Derivative control is additionally used to smooth the curve between positions, acting as a damping term. This control was implemented by storing the previous error value and numerically computing the derivative of the error with respect to time with the division of the difference in error by the interval. Multiplying this term by K_d yields the voltage requested by derivative control. The three of these gains in conjunction comprise the PID controller commonly used today. Our code is backed up by a similar procedure followed by a paper from James Madison University [24].

6.4 User Interface Development

Professor Xing requested us to develop a graphical user interface, or “GUI,” that students or other end users could interact with to control the feed drive and receive feedback. For this task, we opted to use a built-in MATLAB add-on called App Designer. We determined that the GUI was to be broken into two sections, one for user inputs and one for real-time data collection. The user inputs tab allows our system to maintain its integrity as an educational device as it lets users directly change portions of the code through serial communication. The data-collection tab lets the user see the results of their changes, which they then deem desirable or undesirable. The below figure shows the user interface after a sample closed loop run.



Figure 33: GUI as it will appear in the Controls Lab.

As can be seen in the above figure, the GUI was programmed to allow the user to select the three gains of the PID controller which are then written over the serial port and read/implemented on the backend. The user can also enter a series of three desired positions that the table will attempt to reach. The final input requested from the user is selecting whether the run will happen in open or closed loop.

After the user presses the “enter button,” all the inputs are written over the serial port, the table zeros on the limit switch, and then the real time data collection begins. The two output variables are the tables

linear position and speed, both of which appear on their own set of axes. The source code has also been configured so that after the run is complete, three arrays corresponding to the times, positions, and velocities are exported to the base MATLAB workspace so that the user can save the data for later analysis.

6.5 Challenges/Considerations

One of our main challenges was figuring out how to manufacture the frame. After having a discussion with Professor Schuster and seeking the guidance of Eric Pulse we felt like it was in our best interest to machine the parts ourselves and utilize the shops. This ended up being a very fun experience and through this process, we were able to work as a team and put to practice Cal Poly's Learn by Doing slogan. In addition, Nick De Simone and Juan Majano were able to get their yellow tags and take advantage of the end-mill for the higher precision parts.

One challenge in manufacturing the frame was aligning holes between components. Before we had access to a high-precision mill, we relied on our measurements and a drill press. This forced us to oversize our clearance holes to ensure bolted joint connections between faces.

7. Design Verification Plan

To verify that our feed drive prototype will meet the system requirements in Table 4, our team conducted testing as detailed in the following section. Each specification has been assigned a description, measurement, and acceptance criteria; please note the complete Design Verification Plan in Appendix A.13 that will serve as a thorough checklist at the time of testing.

7.1. Mechanical Specifications

This subcategory was defined by purely mechanical test specifications and included the coworking of the table, ballscrew, linear rails, and linear/thrust/ball bearings. Testing occurred in Ryan and Caleb's San Luis Obispo home with the components that make up our structural prototype, here the team manually verified the required acceptance criteria before powering the system. Per our sponsor's initial problem statement, the feed drive table must carry a 5kg weight and travel up to 30 cm; the linear stage assembly – including the table, linear rails and bearings, and ballscrew and ballnut – must produce linear motion at the table through rotary motion of the ballscrew. Thus, the system must concurrently (1) match these acceptance criteria (2) withstand the 5 kg load without deflection and (3) travel 30 cm unimpeded, each of which were verified by physical measurement with dial calipers and measuring tape. We also ascertained the resolution of our motion system, which was accomplished at the time of observing its rotary to linear motion conversion. As an acceptance criterion, the table should move 15 cm for 30 manual turns of the ballscrew. For the last of our manual specifications, we verified that the system is portable – a design request by our sponsor for use in the classroom as context; our current acceptance criterion lists a maximum system weight of 50 lb, where the total system was placed on a scale and measured against this value.

7.2. Electromechanical Specifications

Our system inherently requires electromechanical operation – involving the servomotor, microcontroller, and user interface – and demands consequent performance specifications. Of note, the motor and microcontroller must communicate effectively with one another, as prompted to operate by a frontend computer user interface. We have specified that command prompts from the UI must travel via serial communication to the backend, which we tested at Ryan and Caleb's home. Because our system will operate in the educational setting, its user interface should be easily navigated; an acceptable level of understanding was determined by asking an engineering roommate to operate the interface and run the system in less than

five minutes. Regarding motor performance, we needed to observe as to whether the motor spindle rotation will produce a table travel speed of 10 cm/s, verified by timing the table's end-to-end travel as well as the speed recorded by the controller in software, as well as limit the table's travel length to a maximum of 30 cm (necessitating limit switches that communicate with the microcontroller and stop motion when the maximum distance has been reached). Concurrent to verification of these specifications, our code collects data in real time, performing the recording and storing of motor speed and table position, speed, and error over time.

7.3 Test Procedures Conducted

The following table contains a summary of the key results from our testing phase. Complete information on each of the testing procedures and their full results can be found in the DVP&R in Appendix A.13.

Table 9: Testing Results Summary

Test Name	Results	Notes/Recommendations
Table Travel Test	25 cm	<ul style="list-style-type: none"> Table only able to travel 25cm due to physical constraints. This translates to <25cm on the GUI side due to overshoot. Recommendation: Increase system length or decrease width of the table.
Microcontroller Communication Test	Pass	<ul style="list-style-type: none"> When switches are pressed during the run function, they stop the table immediately.
Load Requirement	Pass	<ul style="list-style-type: none"> No deflection could be measured with a caliper.
Table Speed Test	7.5 cm/s	<ul style="list-style-type: none"> Under peak conditions, the table has a max speed of 7.5 cm/s. Recommendation: Get a more powerful motor or higher quality bearings to minimize losses.
Mobility Test	24.8 lb	<ul style="list-style-type: none"> System weighed on a scale.
Open Loop Motor Output Test	Pass	<ul style="list-style-type: none"> Table speed varies linearly with increasing duty cycle.
Encoder Test	.29 cm	<ul style="list-style-type: none"> Average bias error from three runs

		<ul style="list-style-type: none"> • Recommendation: recalibrate and decrease the speed of the table during a zeroing function to reduce overshoot
User Interface Test	29.3 s	<ul style="list-style-type: none"> • Average time for 3 inexperienced users to design controller and run system using final iteration of MATLAB GUI.
Motion System Resolution	15.00 cm	<ul style="list-style-type: none"> • Confirmed ballscrew pitch using caliper over 3 trials.

Detailed in the following sections of Chapter 7 are brief descriptions of each test that was conducted on the CNC Feed Drive.

7.4 Table Travel Test

The purpose of this test was to verify that the table could travel 30 cm along the ball screw’s longitudinal axis per our sponsor’s product specification. Also, the table must be able to do so without impedance by misalignment or any other failure. This was done by manually turning the ball screw to verify that it produced linear motion of the table. Then marking the table’s center position by placing the tape on its side using a marker to scribe the table’s center on the tape. After that, we aligned the table and the tape to begin running the system. We stopped the ball screw when it was not able to travel any further. This test was performed in the house of two team members. The results were that the table was able to travel 25 cm due to physical constraints. This test failed based on our criteria and some recommendations we have are to increase the system length or decreases the width of the table.

7.5 Microcontroller Communication Test

Before the full initialization of the system, we verified the functionality of all electrical components in tandem with the microcontroller. This included the controller’s ability to execute MicroPython Code, communicate with a front-end computer via the serial port, and interface with the motor, encoder, and limit switches. To apply sufficient power to the motor, we used a Motor Driver breakout board stacked on the Nucleo, which required additional testing.

After the completion of the three hardware drivers for the Nucleo, consisting of the Limit Switch Driver, Motor Driver, and Encoder Driver, we uploaded all three files to the controller for testing. We connected the four encoder pins and input the Counts per Revolution (CPR) of the encoder to our driver. We verified that one mechanical rotation of the motor spindle equated to a 360-degree revolution in software. After connecting our two motor output pins and enabling the motor via a third pin, we successfully ran 12V into the motor, which we verified with a multimeter. We additionally successfully flipped the polarity of the leads in software, which resulted in the motor spinning in the opposite direction. We tested our limit switch implementation by conducting a continuity test with a multimeter, verifying the circuit as closed when not engaged and open when engaged. After testing each component, we constructed the system to test the three pieces of hardware in parallel with a script called “Testcode.py”, which successfully ran the table back and forth between the two limit switches and printed the encoder readings. This verified

system performance and sufficient power draw from the controller. To decrease power consumption, we switched from using optical limit switches to mechanical switches.

We finally tested our serial communication through the initial test code and then verified our system with our closed-loop control implementation. We successfully passed PID controller gains and reference positions to the controller from the front end and passed back time, position, and velocity in real-time.

7.6 Load Requirement Test

The load test was designed to test the feed drive's capability to withstand the rated payload. As set forth by Dr. Xing, we designed our system to effectively carry 10lb on the table. We began by moving the table to the center of the ball screw at which point maximum deflection could occur. We then measured the vertical distance between the baseplate and the top of the table. Afterward, we stacked a 10lb weight atop the table and remeasured the distance. No deflection occurred that was large enough to be measured by a caliper with a resolution of .0001 inches. Therefore, it could be said that our maximum deflection was .0001 inches, and the test was passed.

7.7 Table Speed Test

To determine the top speed of our table, we ran our system in closed loop via our front-end user interface, requesting the furthest possible distance of 25cm and using a Proportional gain of 10 to ensure full saturation of the controller. This saturation ran the motor at the maximum possible voltage of 12V for the greatest possible distance within the actual context of how our system will be run in the lab. Our encoder driver file contains built-in tested functionality to convert the observed RPM of the motor spindle to the linear speed of the ball nut in cm/s via the known pitch of the ballscrew. In observing the plot of velocity against time on the front-end, we observed a maximum linear velocity of 7.5 cm/s. While this is less than the 10 cm/s outlined in our requirements document, we still believe this is sufficient for a satisfactory educational experience centered around linear actuation. If a higher top speed is desired, we recommend the purchase of a more powerful motor or higher quality drivetrain components such as the bearings to minimize losses due to friction.

7.8 Mobility Test

To function as a classroom model, our sponsor requested that the feed drive be portable. In turn, we designed the system to be less than 50 pounds in weight as this was determined a reasonable weight for one person to carry the system as needed. Thus, the system needed to weigh less than 50lb to satisfy the pass criterion of this mobility test, and we found, by using three springs with known constants and measuring their deflection while carrying the feed drive, that the system weighed 33.2lb, thus completing the mobility test.

7.9 Open-Loop Motor Output Test

The open-loop motor output test was conducted to verify effective communication between the motor and the microcontroller. We needed to ensure that the linear speed of the table varies in a controlled manner per the system's characteristic Pulse Width Modulation (PWM) Duty Cycle. To do so, the output test was conducted at Team F12 Operation Base (Ryan and Caleb's house), where variable voltages were sent to the motor then recorded along with motor output speed in real-time on the Matlab GUI. Our pass criteria were designated as the motor spinning at 1200 rpm, which was found to be true, along with the determination that table speed varies about linearly with increasing voltage duty cycle.

7.10 Encoder Test

For this test, we investigated the accuracy of our encoders with respect to our limit switch datum. For this test we ran the system in closed loop to a requested distance of 5cm from the limit switch. This distance

was measured by converting the measured rotation of the shaft to linear position by multiplying by the pitch. We verified the pitch of the ballscrew in the test described in section 7.12. After running our system in closed loop to within 0.5mm of our requested 5cm distance, we measured the distance from the switch to the side of the ballnut with a dial caliper. We conducted 5 tests in a row to observe the travel accuracy of the ballscrew and we observed a clear bias error, as seen in our results in Table 10 below:

Table 10: Measured distance from the datum after the encoder displays 5cm of actuation on the computer.

Run Number	Measured Distance from the Datum [cm]
1	5.34
2	5.20
3	5.33

On average, we see an average position of 5.29 cm when 5 cm is read by the encoders. While additional testing shows very low uncertainty between two non-zero positions, such as 5 cm and 10 cm, this 0.29 cm bias is primarily due to our use of an “imaginary” datum. We define our datum in software to be the contacted limit switch when closed. Since it is difficult to consistently obtain the same datum when the switch closes, we observe that this test fails with respect to a datum, even though motion between two non-zero points has better than 50-micron accuracy. We recommend for optimization to be made with respect to the ballnut slowing down while approaching the limit switch to prevent overshoot into the switch.

7.11 User Interface Test

The purpose of the user interface test was to first test the reliability of the serial communication between the backend code and frontend code and secondly ensure that a user could run the system in a reasonable amount of time, given proper instruction.

To collect valuable data, we selected three engineering students who had not previously seen the interface and who had little experience designing controllers. These people were thought to be representative of the lower 25% of users in the control lab. I then individually explained the procedure to each of the testers without a demonstration and in a similar manner to the prelab briefings that are commonplace in the lab. The three testers then separately entered all their inputs and ran the system.

The average time between touching the keyboard and running the system was 29.3s. We thought this to be extremely reasonable and even a bit faster than some of the other GUIs that we have interacted with during controls labs. The subjects also offered great verbal feedback including saying that the interface was “easy to use” and “intuitive” as well as several compliments on the real-time data collection.

7.12 Motion System Resolution

This test consists of verifying the manufacturer's ball crew pitch of .5 cm by manually turning the ball screw 30 times and measuring the distance it traveled. This distance should be around 15 cm. This test was performed in the house of two team members and the tools consisted of a caliper and the system itself. Two trials were conducted, and the table was able to reach 15 cm both times. As a result, we were able to verify that the manufacturer's listed balls crew pitch was highly accurate. Based on our criteria this test was able to pass.

7.13 Future Testing

The plan going forward is to utilize as much of the software code to further improve upon the mechanical components and testing results. We think that acquiring higher quality parts could increase the table travel speed to be close to 10 cm/s.

8. Project Management

We began our project design by identifying Professor Xing's requested product, a CNC feed drive. Through problem definition, familiarization with system components, and background research, we established a visual on our sponsor's desired functionalities, as well as existent product designs and usages. Through our House of Quality QFD - as detailed in the "Objectives" section - and initial development of a team Gantt chart - as seen in Appendix C - we have outlined our customer's needs and how to accomplish them through design specifications, as seen in Table 3.

We carried out ideation and generated conceptual feed drive models, with emphasis on incorporating the customer needs and wants into a physical model. From there, we evaluated concept prototypes on their ability to satisfy the design specifications and assigned a preliminary design direction (including system components and their layout) as we move into design analysis and part selection. Please see extensive concept design details in Section 4, entitled "Concept Design".

Table 11: Key Deliverables

Deliverable	Deadline
Conceptual Models	10/27/2020
Concept Prototype	11/3/2020
Preliminary Design Review	11/10/2020
Interim Design Review	1/14/2021
Structural Prototype	1/26/2021
Drawing & Manufacturing Plan	2/4/2021
Critical Design Review	2/9/2021
Safety Review	2/18/2021
Manufacturing & Test Review	3/11/2021
Final Design Review	5/25/2021
Project Expo	5/28/2021
Deliver Prototype & FDR	6/3/2021

Fall

Our preliminary design arose from concept selection – a methodical determination of appropriate model features, functions, and components. Critical features of our design include table travel distance (30 cm), accepted carrying load (5kg), product cost (preferably \$300), and service as a portable, educational model. We have modeled each design specification in our concept prototype and have a maximum budget of \$800 to achieve a durable and effective functional prototype.

Research into existent products informed us that equipment which accomplishes such requested system performance was outside of our stipulated budget goal. Therefore, we recognized that specific effort was needed to achieve the durable, low-cost prototype to satisfy our design specifications. Our team received

additional funding from Sponsor Xing for purchase of an apt motor, where preliminary design has informed our decision for each component's form and function. Following PDR submission, our team initiated a bill of materials (BOM) as parts were selected, which prompted their analysis and system-level integration; this work carried into winter as our indented Bill of Materials (iBOM, see Appendix A.11).

Winter

Continued design work included final sizing, selection, and finite element analysis of all system components, as well as completion of the iBOM (Appendix A.11) and modeling of proposed components (Appendix A.12). Our major parts were ordered and delivered in winter - namely the table, ballscrew, ball and thrust bearings, and the linear rails and bearings, all of which comprised our structural prototype. Construction, controlling, and testing of the structural prototype took place as well and informed our team of updated manufacturing needs, including bearing press-fits onto the ballscrew and an altered frame/table design from PLA into aluminum. The Critical Design Review and Report were completed as of February 11, 2021 and carried into manufacturing and risk assessment. Planning and testing for safety, manufacturing, and performance were undertaken as our team evaluated our final design going into Spring Quarter.

Spring

Project work in spring was centered around our final product – including a fully-operational prototype, a front-end MATLAB Graphical User Interface and back-end drivers/controllers, the Final Design Report, and End User Manual all delivered at the Final Design Review on June 3, 2021. Testing into such necessary parameters as mechanical/electrical performance and safety were conducted to confirm our sponsor's needs were met, and the associated CAD and front/back-end source codes were packaged as a deliverable product, ultimately leading to the delivery of this and our final prototype to Professor Xing on June 3, 2021.

9. Conclusion

This Final Design Report outlined our background research through to concept design and selection and ultimately to the manufacturing and testing of our final “CNC Feed Drive” prototype. Our customer's needs were addressed through the purchase and assembly of our servomotor-driven linear guide, with such characteristic components as the ballscrew and ballnut, linear rails and bearings, and table with plate loads. Along with the physical prototype, we delivered a back-end PID controller and front-end MATLAB GUI to control the table's linear motion and provide real-time data collection for lab students.

Overall, our prototype was able to meet all design specifications (as indicated in Table 9 of Chapter 7) except for that of total table travel distance and maximum table travel speed, where the aim was 30cm travel and 10cm/s speed and the actual spec was 25cm and 7.5cm/s, respectively. Much of our difficulty came with component implementation, especially in manufacturing our aluminum frame, table, and rail spacers to a tolerance required for the precision of our system, where the greatest observed detriment was in energy losses from friction and component misalignment. In the face of our inexperience, though, we gained much insight into precision manufacturing and product assembly, having now accumulated more machining hours than we likely imagined. We would also like to note a determined encoder bias, namely that an average positional error of 0.29cm is produced when the table returns to its datum location.

To resolve these shortcomings in future prototypes, we recommend the following solutions: (1) to increase the ballscrew length or decrease the table width, allowing for greater table travel distance (2) to implement a motor with higher power rating to minimize frictional losses, thus increasing table travel speed (3) to build any manufactured parts through the highest precision means possible, such as using a CNC mill or

ordering parts to be built by an experienced manufacturer and (4) to program a “zeroing” function in which the table slows as it returns to its datum position, effectively decreasing the amount of encoder bias produced from overshoot. We see these to be the next steps for creating an effective classroom model and hope to hear of a feed drive implemented in future Cal Poly lab experiments. All in all, it has been an honor and a joy to develop this prototype for Professor Xing, and we are extremely grateful for the experience.

References

- [1] Rivers, Alec Rothmyer, and Llan Ellison Moyer. "Automatically Guided Tools" Patent US10067495B2. 2016.
- [2] Iwashita, Yasusuke, and Satoshi Ikai. "Servomotor Controller for Controlling a Servo Motor Designed to Drive the Feed Axis in a Machine Tool". FANUC Corp, assignee. Patent DE102013103341B4. 20 Nov. 2014.
- [3] Ren, Qian, Bao, Zheng, Wu. "High-speed High- Precision Servo Linear Motor Sliding Table". Patent CN101844314A. 29 Sept. 2010.
- [4] Aoki, Yasuo. "Exposure Apparatus, Movable Body Apparatus, Flat-panel Display Manufacturing Method, and Device Manufacturing Method". Patent KR101911717B1. 25 Oct. 2018.
- [5] Kim, Hounng Joong, and Shigeru Shinohara. "Power Tool with a Linear Motor". Koki Holdings Co Ltd, assignee. Patent US6705408B2. 16 Mar. 2004.
- [6] Aerotech Stages Catalog. (n.d.). Retrieved October 12, 2020, from <https://www.aerotech.com/productcatalog/stages.aspx>
- [7] Motorized Linear Stages. (n.d.). Retrieved October 12, 2020, from <https://www.newport.com/c/motorized-linear-stages>
- [8] Motorized X-Axis Stages. (n.d.). Retrieved October 12, 2020, from <https://us.misumi-ec.com/vona2/mech/M0500000000/M0506000000/M0506030000/?searchFlow=results2category>
- [9] Linear Actuators. (n.d.). Retrieved October 12, 2020, from <https://www.grainger.com/category/power\transmission/linear-motion/linear-actuators?attrs=Rated+Load%7C27+lb>
- [10] Our Lifts: Material Handling Equipment: Custom Lifting Devices. (2020, February 13). Retrieved October 12, 2020, from <https://alum-a-lift.com/our-lifts/>
- [11] McMaster Carr Hydraulic Cylinders. (n.d.). Retrieved October 12, 2020, from <https://www.mcmaster.com/hydraulic-cylinders/>
- [12] Erkorkmaz, Kaan & Altintas, Yusuf. (2001). "High speed CNC system design. Part II: Modeling and identification of feed drives". *International Journal of Machine Tools and Manufacture*. 41. 1487-1509.
- [13] Verma, Kuldeep, and R.m. Belokar. "Inclusive Estimations of Ball Screw-based CNC Feed Drive System over Positioning and Pre-loading Factor." *Assembly Automation* 38.3 (2018): 303-13.
- [14] Breaz, Radu Eugen, Sever Gabriel Racz, Octavian Bologna, and Melania Tera. "Model of a CNC Feed Drive for On-Site Tuning of the Controllers for Single Axis Motion." *Applied Mechanics and Materials* 841 (2016): 133-38.
- [15] Low, Kay-Soon & Keck, Meng-Teck. "Advanced Precision Linear Stage for Industrial Automation Applications." *IEEE: Transactions on Instrumentation and Measurement* 52 (2003): 785-89.

- [16] Breaz, Radu Bologa, Octavian Oleksik, Valentin Racz, Gabriel. “Computer Simulation for the Study of CNC Feed Drives Dynamic Behavior and Accuracy.” *Proceedings, Eurocon 2007 – The International Conference on Computer as a Tool (2007)*.
- [17] Wai, R.-J. “Adaptive Sliding-Mode Control for Induction Servomotor Drive.” *IEEE Proceedings Electric Power Applications*, vol. 147, no. 6, Nov. 2000, pp. 553–562.
- [18] Zhang, Ya Wei, and Wei Min Zhang. “Vibration Analysis of Ball Screw Drive System for CNC Machine Tool.” *Advanced Materials Research*, vol. 139-141, 2010, pp. 1224–1228.
- [19] Basics of Feed Drive in CNC machine. (n.d.). Retrieved October 12, 2020, from <http://toolingworld.com/basics-feed-drive-cnc-machine>
- [20] Technical Note. (n.d.). Retrieved October 12, 2020, from <https://www.newport.com/n/stagecomponents-considerations>
- [21] North American Industry Classification System, Code Descriptions (335312 – Motor and Generator Manufacturing). Retrieved October 12, 2020, from <https://www.naics.com/naics-codedescription/?code=335312>
- [22] United States Department of Labor, Regulations (Standards - 29 CFR). Retrieved October 12, 2020, from <https://www.osha.gov/laws-regs/regulations/standardnumber/1926>
- [23] Farah, Shady, et al. “Physical and Mechanical Properties of PLA, and Their Functions in Widespread Applications — A Comprehensive Review.” *Advanced Drug Delivery Reviews*, vol. 107, Dec. 2016, pp. 367–92. From: https://dspace.mit.edu/bitstream/handle/1721.1/112940/Anderson_Physical%20and%20mechanical%20properties.pdf?sequence=1&isAllowed=y
- [24] Prague, Nathan “Controlling Physical Systems”, 2016, From: https://w3.cs.jmu.edu/spragunr/CS354_F16/handouts/pid.pdf

Appendices

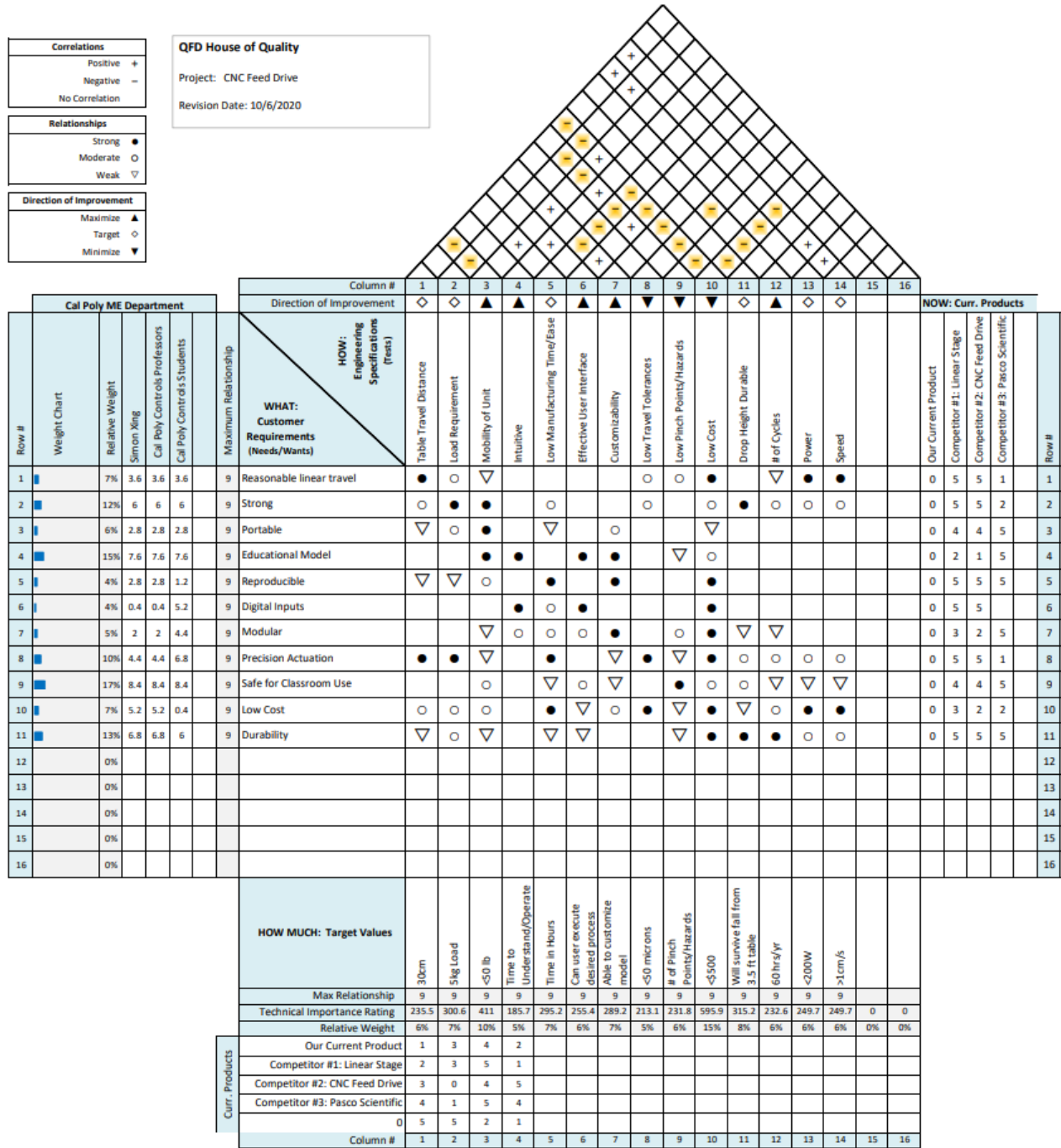
- A.1. Pairwise Comparison Chart
- A.2. QFD House of Quality
- A.3. Gantt Chart
- A.4. Jamboard Ideation Session
- A.5. Ideation Models
- A.6. Additional Concept Prototype Views
- A.7. Pugh Matrices
- A.8. Weighted Decision Matrices
- A.9. Preliminary Power Analysis
- A.10. Preliminary Power Analysis
- A.11. Indented Bill of Materials
- A.12. Failure Modes and Effects Analysis (FMEA)
- A.13. Design Verification Plan (DVP&R)
- A.14. Design Selection Analysis
- A.15. Drawing Package
- A.16. Risk Assessment
- A.17. Electrical Schematics/Wiring Diagram
- A.18. Annotated Micro-controller code
- A.19. Test Procedures
- A.20. User Manual
- A.21 MATLAB GUI Code

A.1. Pairwise Comparison Chart

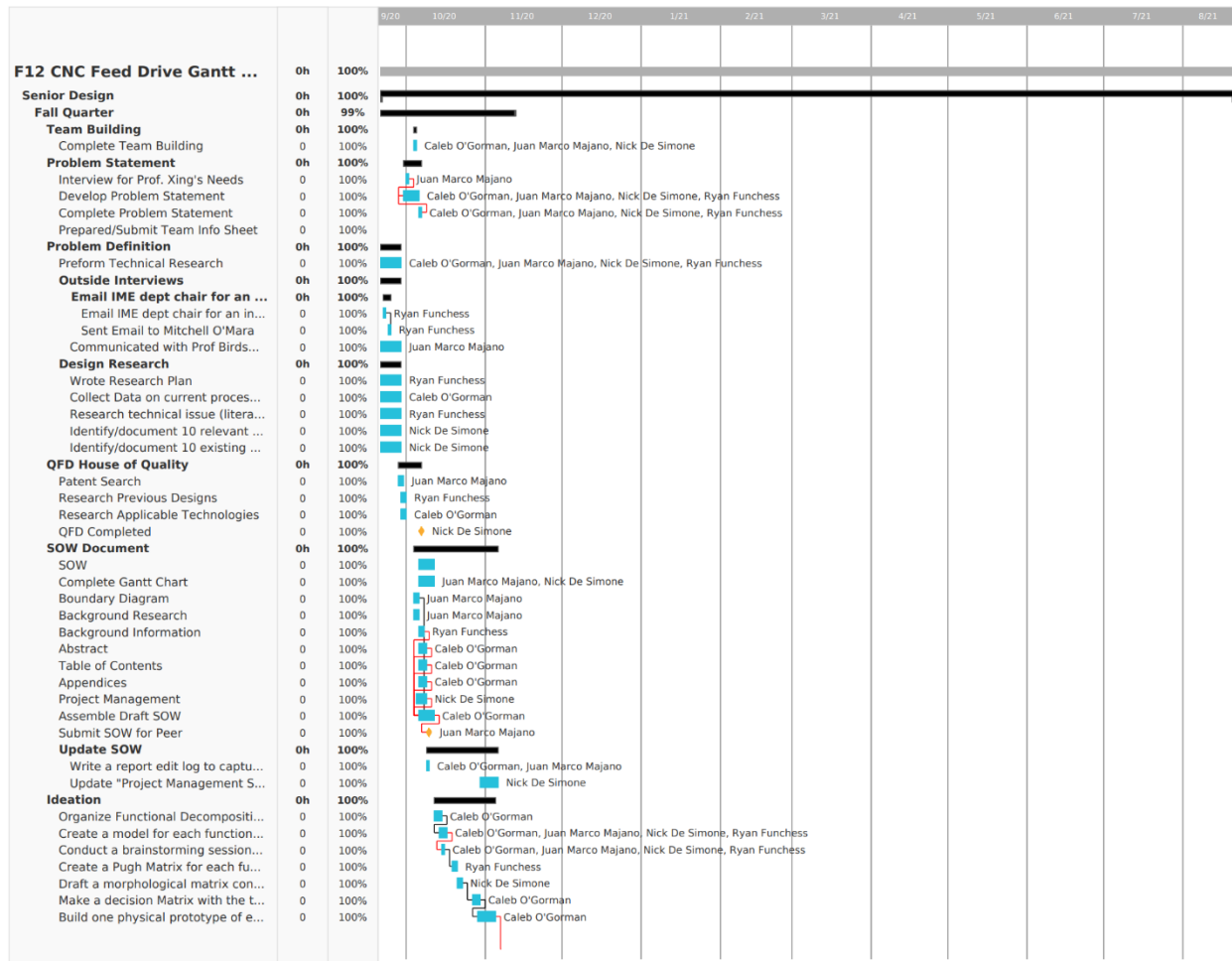
Professor Pairwise	Reasonable	Strong	Portable	Educational	Reproduci	Digital	Modular	Precision	Safe for	Low Cost	Durability
Reasonable linear travel	0.5	1	0	1	1	0	0	0	1	1	1
Strong	0	0.5	0	1	0	0	0	1	1	0	1
Portable	1	1	0.5	1	0	0	1	1	1	1	1
Educational Model	0	0	0	0.5	0	0	0	0	1	0	0
Reproducible	0	1	1	1	0.5	0	0	1	1	1	1
Digital Inputs	1	1	1	1	1	0.5	1	1	1	1	1
Modular	1	1	0	1	1	0	0.5	1	1	1	1
Precision Actuation	1	0	1	1	0	0	0	0.5	1	1	1
Safe for Classroom Use	0	0	0	0	0	0	0	0	0.5	0	0
Low Cost	0	1	0	1	0	0	0	0	1	0.5	1
Durability	0	1	0	1	0	0	0	0	1	0	0.5
Sum	4.5	7.5	3.5	9.5	3.5	0.5	2.5	5.5	10.5	6.5	8.5
Normalized Sum	0.072	0.12	0.056	0.152	0.056	0.008	0.04	0.088	0.168	0.104	0.136
%	7.2	12	5.6	15.2	5.6	0.8	4	8.8	16.8	10.4	13.6

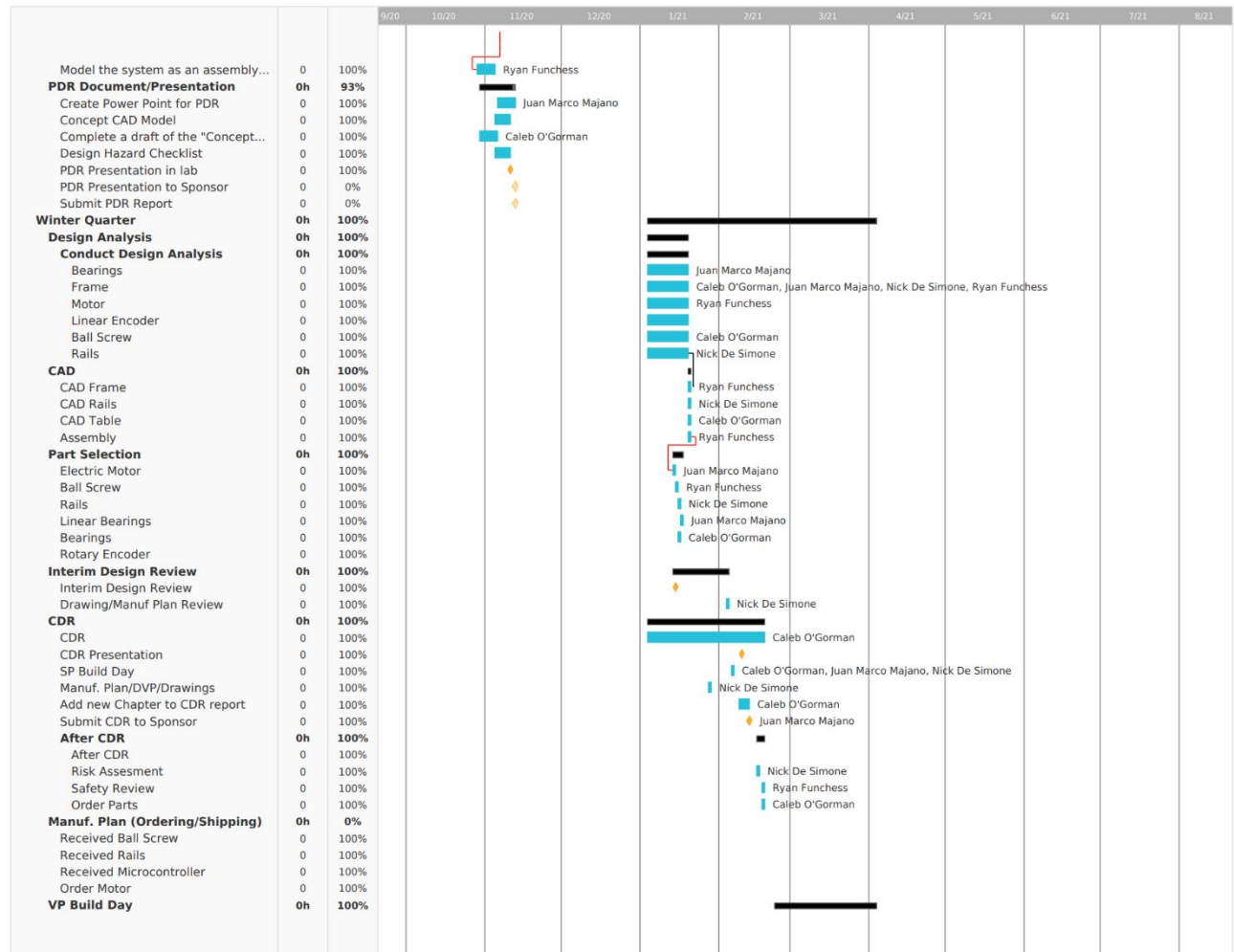
Student Pairwise	Reasonable	Strong	Portable	Educational	Reproduci	Digital	Modular	Precision	Safe for	Low Cost	Durability
Reasonable linear travel	0.5	1	0	1	0	1	1	1	1	0	0
Strong	0	0.5	0	1	0	0	0	1	1	0	1
Portable	1	1	0.5	1	0	1	1	1	1	0	1
Educational Model	0	0	0	0.5	0	0	0	0	1	0	0
Reproducible	1	1	1	1	0.5	1	1	1	1	0	1
Digital Inputs	0	1	0	1	0	0.5	0	1	1	0	1
Modular	0	1	0	1	0	1	0.5	1	1	0	1
Precision Actuation	0	0	1	1	0	0	0	0.5	1	0	1
Safe for Classroom Use	0	0	0	0	0	0	0	0	0.5	0	0
Low Cost	1	1	1	1	1	1	1	1	1	0.5	1
Durability	1	1	0	1	0	1	1	1	1	0	0.5
Sum	4.5	7.5	3.5	9.5	1.5	6.5	5.5	8.5	10.5	0.5	7.5
Normalized Sum	0.072	0.12	0.056	0.152	0.024	0.104	0.088	0.136	0.168	0.008	0.12
%	7.2	12	5.6	15.2	2.4	10.4	8.8	13.6	16.8	0.8	12

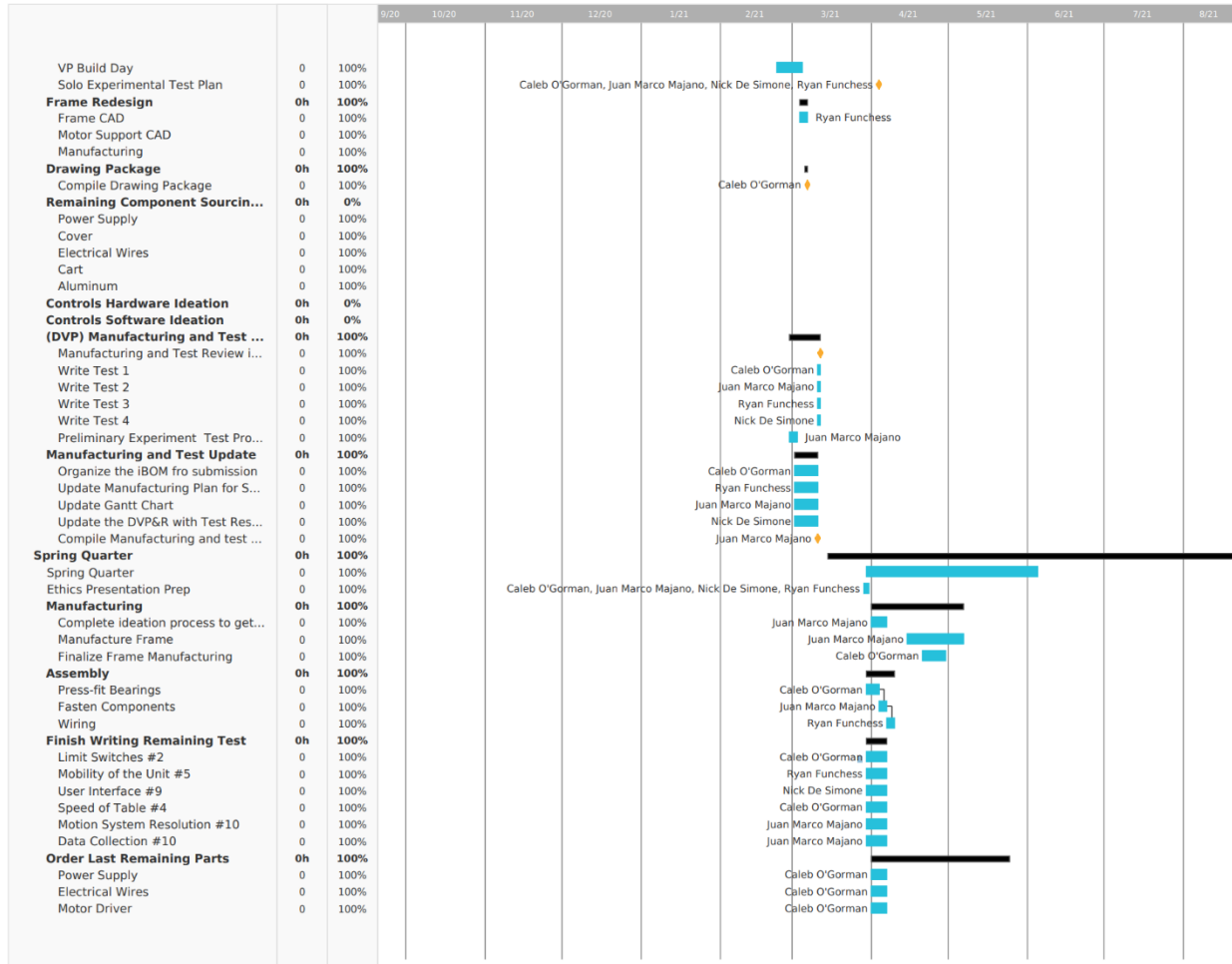
A.2. House of Quality

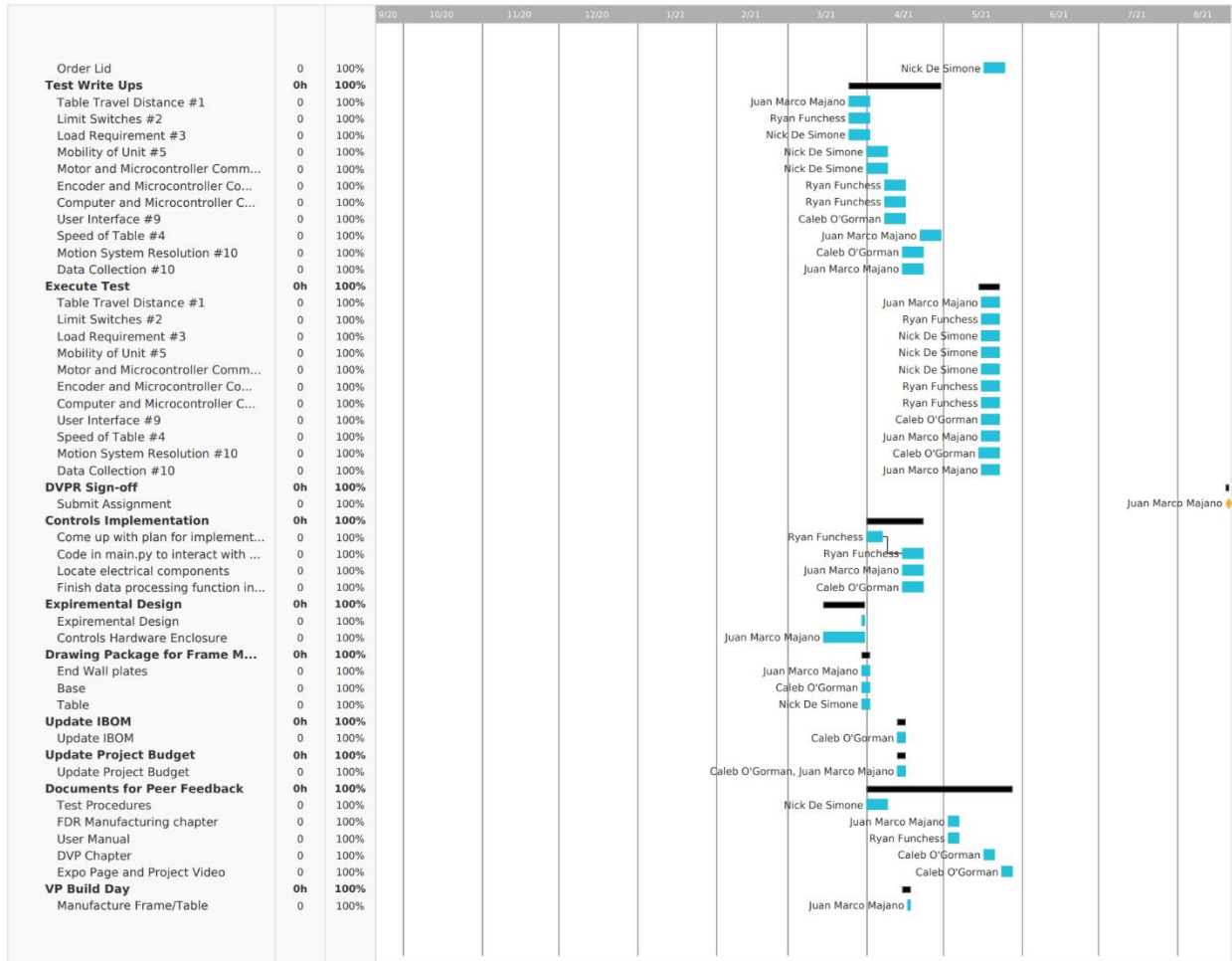


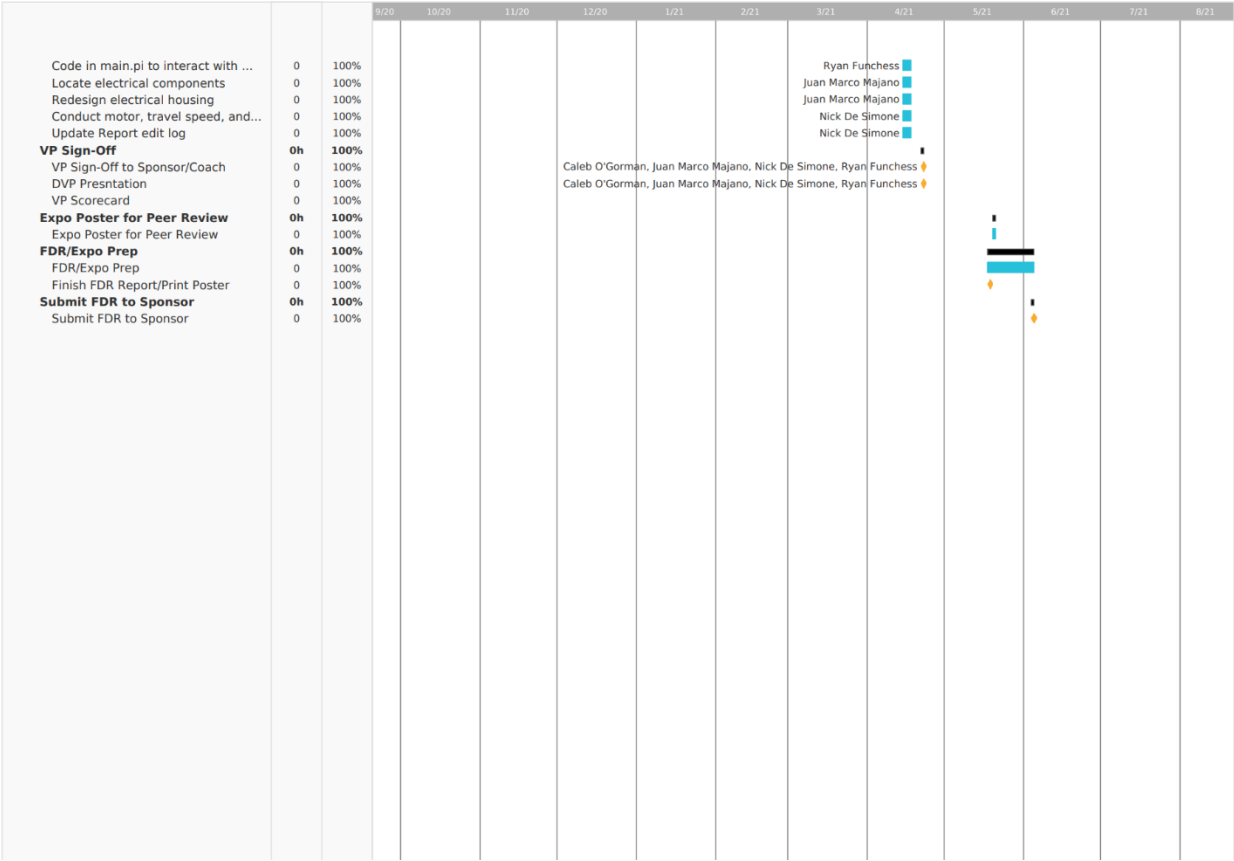
A.3. Gantt Chart







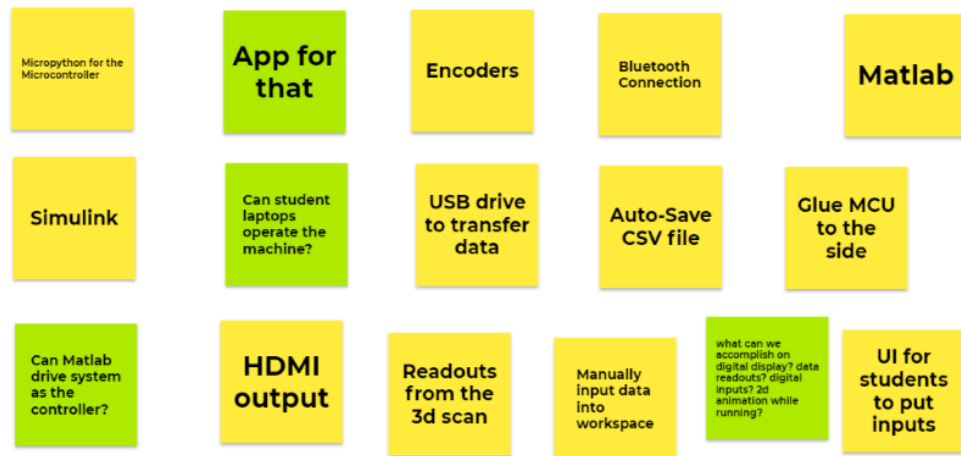




A.4. Jamboard Ideation Session

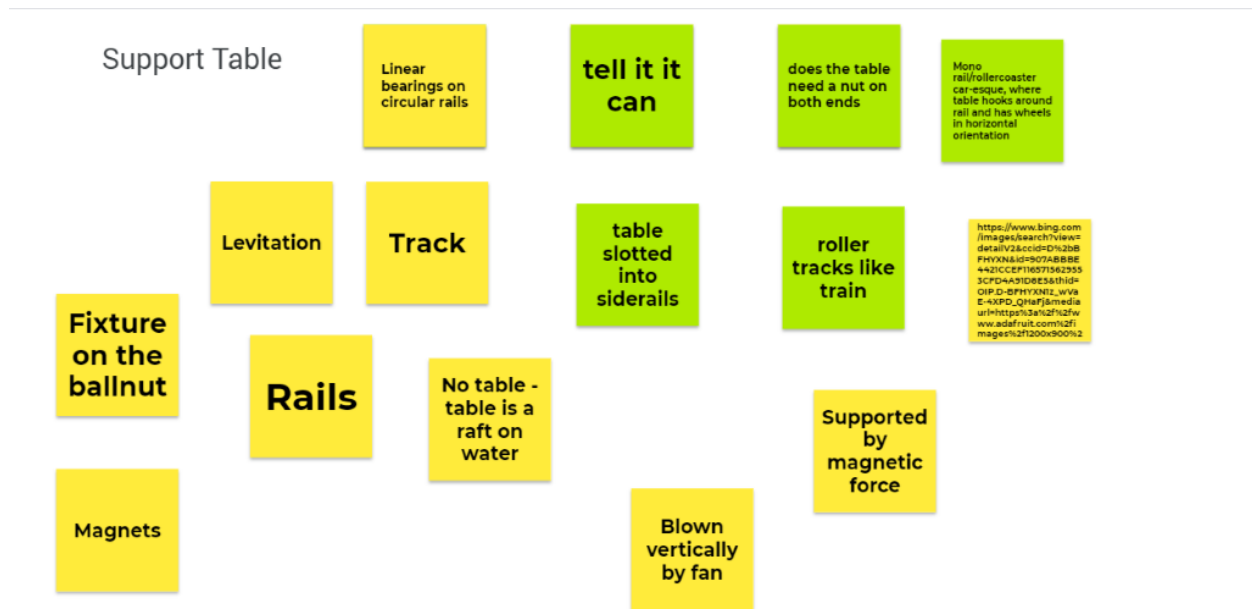


Interact with Student Software

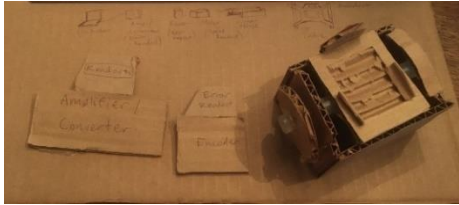


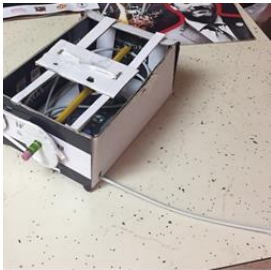



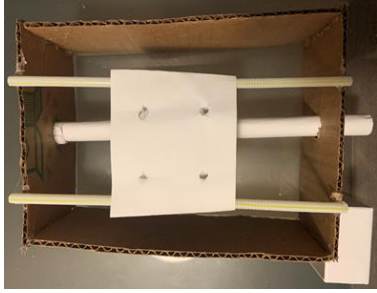
Teach Students



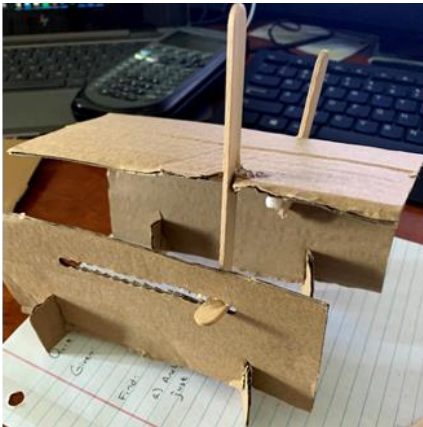


A.5. Ideation Models

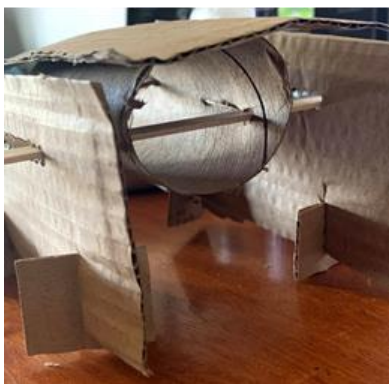
	<p>Model N.1: Overall system diagram, including table, ball screw, and readouts.</p>
	<p>Model N.2: Top view of rolling cart assembly, where each system component should be individually visible/manipulatable.</p>
	<p>Model N.3: Inset load-carrying table to hold load in place (divots, raised lip or the like).</p>
	<p>Model R.1: Model of the feed drive table, railings, ball screw, gear train and power connection.</p>
	<p>Model R.2: Close-up of motor-ball screw gear train.</p>



Model C.1: Ideation model of feed drive table, ball screw, railings, and motor.



Model J.1: Model of “slot and pin” consideration for supporting the feed drive table.



Model J.2: Ideation model that practices how to support table.

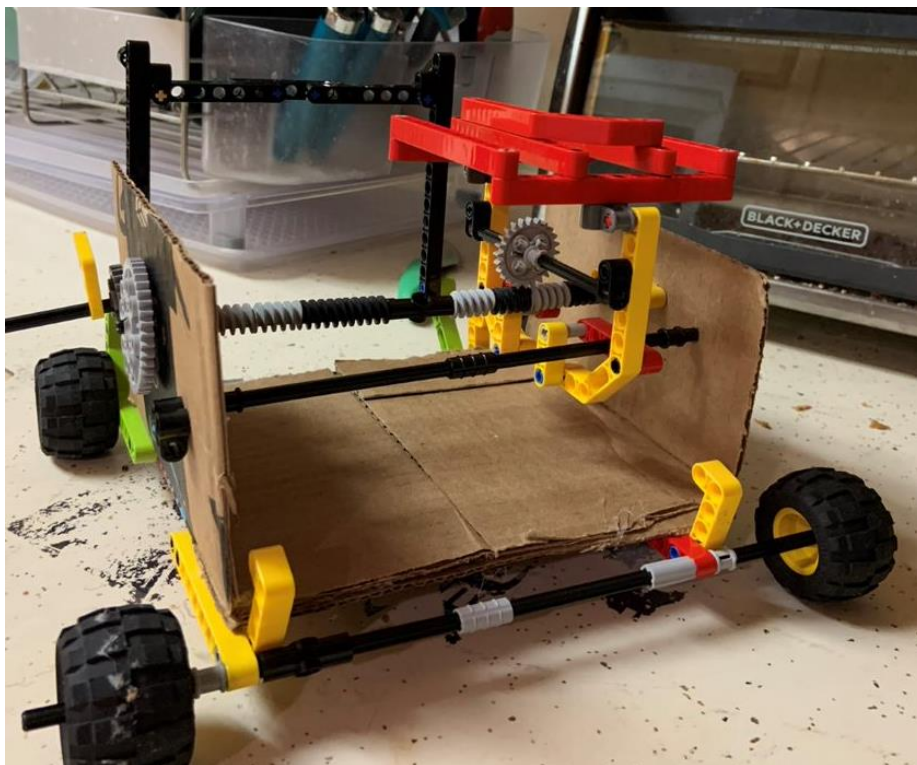
A.6. Additional Concept Prototype Views



Top View



Side View



Additional View

A.7. Pugh Matrices






Pugh Matrix - Support Table

Concept & Active	1. Motors & R	2. Slot & Pin	3. Wheels & mid guard	4. Linear bearing	5. Frictionless surface
Strong	D	+	S	S	S
Reasonable Linear Travel	A	S	S	S	-
Precision Actuation	T	-	S	+	-
Safe for classroom use	V	-	+	S	+
Low cost	M	+	+	-	+
Durability	-	+	S	+	-
Total	0	1	2	1	-1

	Rock and Ratchet	Pugh matrix Concepts Ball screw	Hydraulic Act.	Ball and Ratchet	Win
Datum	-	-	+	-	-
Option	+	+	-	+	-
Datum	S	-	+	S	S
Datum	S	S	S	S	-
Datum	+	-	-	-	-
Datum	S	+	+	+	-
Datum	S	+	+	+	-
Datum	+	-	-	-	-
0	2	-3	1	-	-

	1) Switchcase	2) Brief Case	3) Removable Cart	4) Fixed Cart
Reasonable Linear Travel	S	S	Σ	S
Strong	S	S		S
Portable	+	+		-
Educational	S	S		-
Reproducible	-	-	T	S
Digital Inputs	S	S		S
Modular	S	S		S
Precision Actuation	S	S		S
Safe	+	+	D	+
Low Cost	S	S		S
Durable	+	+		+
Totals	+2	+2		0

Criteria	1) Caliper	2) Draw Wire	3) Linear Encoder	4) Rotation Encoder	5) Step Counter
Reasonable Linear Travel	-	S	S	DATUM	S
Strong	S	S	S		S
Portable	-	-	S		S
Educational	-	-	S		-
Reproducible	S	-	S		S
Digital Inputs	-	-	S		S
Modular	S	S	S		S
Prec Actuation	+	-	+		-
Safe	-	S	S		S
Low Cost	S	+	-		S
Durable	+	+	+		S
Totals	-3	-3	+1		-2

Concept	① Screws	② VICE	③ V-Block	④ V-Table	⑤ L-Bracket
Criteria					
Reasonable Linear Travel	DATUM	-	S	-	-
Strong		S	S	-	-
Portable		-	-	-	-
Reproducible		-	S	-	-
Modular		-	S	-	-
Safety		S	-	-	-
Low Cost		-	+	-	-
Durability		-	+	+	-
TOTAL		-5	0	-7	-8

A.8. Weighted Decision Matrices

Move Table		Rack and Pinion		Ball screw		Design 3: Belt and Pulley		Design 4: Conveyor Belt	
Criteria	Weights	Score	Total	Score	Total	Score	Total	Score	Total
Reasonable Linear Travel	7	3	21	1	7	2	14	3	21
Strong	12	2	24	3	36	1	12	1	12
Portable	6	3	18	3	18	2	12	3	18
Digital Inputs	4	3	12	3	12	2	8	3	12
Precision Actuation	10	2	20	3	30	1	10	0	0
Safe for Classroom use	17	2	34	2	34	3	51	2	34
Low Cost	7	2	14	2	14	2	14	3	21
Durability	13	3	39	3	39	1	13	1	13
Total			182		190		134		131

Support Table		Monorail		Slot and Pin		Wheels and Rail Guard		Linear Bearing	
Criteria	Weights	Score	Total	Score	Total	Score	Total	Score	Total
Reasonable Linear Travel	7	3	21	1	7	3	21	3	21
Strong	12	3	36	2	24	3	36	3	36
Precision Actuation	10	2	20	1	10	2	20	3	30
Safe for Classroom use	17	2	34	2	34	3	51	3	51
Low Cost	7	2	14	3	21	2	14	0	0
Durability	13	2	26	2	26	2	26	3	39
Educational Model	15	2	30	1	15	2	30	3	45
Total			181		137		198		222

Move System		Suitcase		Briefcase		Removable Cart		Fixed Cart	
Criteria	Weights	Score	Total	Score	Total	Score	Total	Score	Total
Portable	6	2	12	2	12	3	18	1	6
Safe for Classroom use	17	3	51	3	51	3	51	1	17
Low Cost	7	1	7	1	7	3	21	3	21
Durability	13	2	26	2	26	1	13	3	39
Total			96		96		103		83

Secure Load		Screws		Boundaries		Vice		L-Brackets	
Criteria	Weights	Score	Total	Score	Total	Score	Total	Score	Total
Strong	12	3	36	3	36	2	24	1	12
Portable	6	3	18	3	18	2	12	2	12
Reproducible	4	2	8	3	12	2	8	1	4
Modular	5	3	15	1	5	1	5	2	10
Safe for Classroom use	17	3	51	1	17	1	17	2	34
Low Cost	7	3	21	2	14	1	7	1	7
Durability	13	2	26	3	39	1	13	2	26
Total			175		141		86		105

Monitor Position		Linear Encoder		Rotational Encoder	
Criteria	Weights	Score	Total	Score	Total
Reasonable Linear Travel	7	2	14	3	21
Digital Inputs	4	2	8	3	12
Precision Actuation	10	3	30	1	10
Low Cost	7	1	7	3	21
Educational	15	3	45	1	15
Durability	13	3	39	3	39
Total			143		118

A.9. Preliminary Power Analysis

Motor Sizing		
<u>Quantity</u>	<u>Value</u>	<u>Unit</u>
Payload Mass	20	kg
Min Speed	0.01	m/s
Ballscrew Efficiency	0.8	
Motor Projected Efficiency	0.75	
Mean Diameter	19	mm
Pitch	5	mm
Lead	5	mm
Friction Coefficient	0.74	
Friction Force	145.04	N
Torque to Drive	0.14	Nm
Angular Velocity Required	2	rot/s
Angular Velocity Required	120	rpm
Angular Velocity Required	12.57	rad/s
Motor Power	2.42	W

A.10. 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
Rotating machinery	The ballscrew is a rotating component that poses danger when contacted by humans or other objects. Users should wear safety glasses during use and tie back long hair. Users should keep body and foreign objects away from the inside of the system while system runs.	1/26	4/5
Pinch points	Moving components of the system, such as the ballnut, are potential pinch points for users. People must keep bodies away from moving machinery while machinery runs.	1/26	4/26
High accelerations/High force	The CNC Feed Drive Table can undergo high accelerations due to high force when being operated by the designed control system. Users must keep bodies away from moving machinery while machinery runs.	1/26	5/3
Potential to fall	The CNC Feed Drive System is comprised of significant mass and can cause damage if dropped. The final design will include handles, or a similar technology that will allow ease of simple transport. Additionally, the cart that will be used to transport the system will have the capability to secure the system using straps or a similar technology.	2/9	5/3
Vibration	Running the ballscrew at resonant frequency can create unstable vibration and high amplitude of oscillation. The ballscrew will be sized for a natural frequency well above our system's operating speed, and a limit will be placed in the Software regarding the speed that the ballscrew can operate at.	1/14	5/17
High voltage/Extreme Position/Control System Malfunction	If the control system or a piece of hardware malfunctions, limits must be put in place to prevent unstable operation. At the extreme positions, a limit switch and kill switch will be implemented to cut system power. Additionally, hard stops can be used as a last case resort. In the case that the control system calls for excessive power to the motor, Saturation Values can be set in the Control Algorithm, which designates values for voltage that will not be exceeded as an output.	2/9	5/24

A.11. Indented Bill of Materials/Project Budget

Team: F12

CNC Feed Drive Indented Bill of Material (iBOM)

Assembly Level	Part Number	Description	Qty	Unit Price	Ttl Cost	Source	More Info
		Lvl0 Lvl1 Lvl2 Lvl3 Lvl4					
0	10000	CNC Feed Drive Assembly				-----	
1	11000	Drive Train				-----	
2	11100	Motor Assembly				-----	
3	11110	Motor w/ Encoder	1	\$46.98	\$46.98	ServoCity	Product ID: 5202-0002-0003
3	11120	M4X0.7 - 12 Socket head	4	\$0.74	\$2.96	Home Depot	Note: Product priced as a 4 pack by vendor
2	11200	Ballscrew Assembly	1	-----	-----	-----	Pressfit ballscrew in bearings prior to bearing inserted in walls.
3	11210	Ball Bearing Assembly	1	-----	-----	-----	
2	11211	WBF-20 End Support	1	-----	-----	Automation4Less	Product ID: FF20 (Included with Ballscrew)
4	11212	M6X1.0 - 16 Socket Head	4	\$1.24	\$4.96	Home Depot	Note: Product priced as a 2 pack by vendor (need 8 screws total)
4	11213	M6X1.0 Nuts	4	\$0.72	\$2.88	Home Depot	Note: Product priced as a 2 pack by vendor (need 8 nuts total)
3	11220	Thrust Bearing Assembly	1	-----	-----	-----	
4	11221	WBK - 20 End Support	1	-----	-----	Automation4Less	Product ID: FK20 (Included with Ballscrew)
4	11222	M6X1.0 - 30 Socket Head	4	\$0.75	\$3.00	Home Depot	Note: Product priced as a 2 pack by vendor
4	11213	M6X1.0 Nuts	4	-----	-----	Home Depot	Note: Price and source included above
3	11230	Ballscrew, 25 mm	1	\$283.67	\$283.67	Automation4Less	Product ID: BSFU2505-0500-FS
2	11300	Ballnut Assembly	1	-----	-----	-----	
3	11310	Ballnut	1	-----	-----	Automation4Less	Product ID: HD25 (Included with Ballscrew)
3	11320	Ballnut Bracket	1	-----	-----	Automation4Less	Included with Ballscrew
3	11330	M5X0.8 - 14 Socket head	6	\$0.59	\$8.26	Home Depot	Note: Product priced as a 2 pack by vendor (need 14 screws total)
2	11400	Flexible Shaft Coupling	1	-----	-----	-----	
3	11410	11/16" Flexible Shaft Coupling Iron Hub	1	\$6.25	\$25.00	McMaster	Product ID: 6408K12
3	11420	1/4" Flexible Shaft Coupling Iron Hub	1	\$6.25	\$6.25	McMaster	Product ID: 6408K12
3	11430	Buna-N Rubber Spider	1	\$3.72	\$3.72	McMaster	Product ID: 6408K73
1	12000	Rails Assembly	1	-----	-----	-----	
2	12100	Rails and Linear Bearings	1	\$29.99	\$29.99	Vevor	Product ID: SBR16 (2 RAILS + 4 L. BEARINGS)
2	12200	M5X0.8-30 Socket Head	8	\$0.48	\$3.84	Home Depot	Note: Product priced as a 2 pack by vendor
2	12300	M5X0.8 Nuts	8	\$0.02	\$1.76	McMaster	Note: Product priced as a 100 pack by vendor
1	13000	Mass support	1	-----	-----	-----	
2	13100	Table	1	-----	-----	custom	Printed from PLA
2	13200	Plate Masses	4	\$17.89	\$71.56	Online Metals	Part #: 10274
2	12200	M5X0.8 - 14 Socket head	8	-----	-----	Home Depot	Note: Price and source included above
2	11212	M6X1.0 - 16 Socket Head	4	-----	-----	Home Depot	Note: Price and source included above
2	13300	M10-1.5 x 50 Bolts	4	\$0.74	\$2.94	Home Depot	Note: Product priced as a 2 pack by vendor
2	13400	M10-1.5 Nuts	4	\$0.63	\$3.80	Home Depot	Note: Product priced as a 3 pack by vendor
1	14000	Structure	1	-----	-----	-----	
2	14100	Base Plate	1	-----	-----	custom	manufactured from 1/4" aluminum sheet metal
2	14200	Rail Spacer	1	-----	-----	custom	manufactured from 1/4" aluminum sheet metal
2	14300	Datum Face	1	-----	-----	custom	manufactured from 1/2" aluminum sheet metal
2	14400	Motor Face	1	-----	-----	custom	manufactured from 1/4" aluminum sheet metal
2	14500	Ball Bearing Face	1	-----	-----	custom	manufactured from 1/4" aluminum sheet metal
2	14600	Motor-Side Leg	1	-----	-----	custom	manufactured from 1/2" aluminum sheet metal
2	14700	Opposite-Side Leg	1	-----	-----	custom	manufactured from 1/2" aluminum sheet metal
2	14800	1/4" Dowel Pin	1	\$0.12	\$0.12	Utturn	
2	14900	1/4"-20 Bolt	1	\$4.15	\$4.15	Home Depot	Note: Product priced as a 10 pack by vendor (Ideally find new)
2	14110	1/4"20 Nut	1	\$0.98	\$0.98	Home Depot	Note: Product priced as an 4 pack by vendor
2	14120	Metal Bracket and Screws	2	\$9.95	\$19.90	Amazon	
2	14130	Plastic Corner Bracket	2	\$0.33	\$0.66	McMaster	
2	14140	#8-32 x .775 Screws	8	\$0.12	\$0.98	Home Depot	Note: Product priced as an 8 pack by vendor
2	14150	#8-32 Nuts	4	\$0.09	\$4.68	Home Deopt	Note: Product priced as an 50 pack by vendor
1	15000	Controls	1	-----	-----	-----	
2	15100	Microcontroller	1	\$14.60	\$14.60	Mouser	Mouser #: 511-NUCLEO-L476RG
2	15200	Limit Switches	2	\$4.50	\$8.99	Amazon	Note: Product priced as a 5 pack by vendor
2	15300	Motor Driver	1	\$12.25	\$12.25	ST	
2	15500	Power Supply	1	\$12.99	\$0.00	Amazon	
2	15400	Wires (Multi-Color)	1	\$10.49	\$10.49	Amazon	Note: May be more than necessary
1	16000	Safety Lid	1	-----	-----	-----	Note: Still being designed

Materials Budget for Senior Project

Title of Senior Project: F12 CNC Feed Drive

Team members: Caleb O’Gorman, Juan Majano, Nick de Simone, Ryan Funchess

Designated Team Treasurer: Caleb

Faculty Advisor: Dr. Peter Schuster

Sponsor: Dr. Siyuan Xing

Quarter and year project began: Fall 2021

Materials budget given for this project: \$800.00

Date purchased	Description of items purchased	Vendor/Part Number	How the material will be purchased?	Current Location of Material	Transaction amount	Shipping/handling/tax estimates
02/02/21	PLA Filament	Amazon	Sponsor	San Luis Obispo	\$ 22.99	\$ 1.78
01/28/21	Ball Screw, Ballnut, and Bearings	Automation4Less/BSFU2505-0500-F5	Sponsor	San Luis Obispo	\$ 262.90	\$ 20.77
02/02/21	Rails and linear bearings assembly	Vevor/SBR16 Rail and SBR16uu Linear Bearing	Sponsor	San Luis Obispo	\$ 29.99	\$ -
02/06/21	M6x0.8mm Socket Head screws and washers for ballnut to table assembly	Ace Hardware	Out of Pcket	San Luis Obispo	\$ -	\$ -
02/02/21	NUCLEO-L476RG (Microcontroller)	STI Instruments/511-NUCLEO-L476RG	Sponsor	San Luis Obispo	\$ 22.59	\$ 4.35
03/03/21	Limit Switches and shaft couplings	McMaster-Carr/	Sponsor	San Luis Obispo	\$ 12.68	\$ -
03/03/21	Aluminum Sheets	Grainger/	Sponsor	San Luis Obispo	\$ 194.82	\$ 44.91
04/06/21	Motor Driver	ST/	Sponsor	San Luis Obispo	\$ 12.88	\$ 5.00
04/13/21	Power Supply	Amazon	Sponsor	San Luis Obispo	\$ 13.99	\$ 1.22
04/13/21	Roller Switches	Amazon	Sponsor	San Luis Obispo	\$ 6.99	\$ 0.61
04/13/21	Motor Driver	ST	Sponsor	San Luis Obispo	\$ 12.88	\$ -
04/26/21	56 Fasteners and Corner Brackets	Ace Hardware	Reimbursement	San Luis Obispo	\$ 33.31	\$ 2.91
04/20/21	Female Bullet Adaptors	Servocity/	Reimbursement	San Luis Obispo	\$ 3.99	\$ -
03/03/21	Motor	Servocity/	Sponsor	San Luis Obispo	\$ 39.99	\$ 6.99
budget: \$ 800.00					Total expenses: \$ 758.54	
actual expenses: \$ 758.54						
remaining balance: \$ 41.46						

A.12. Failure Modes and Effects Analysis (FMEA)

Product: _____ Design Failure Mode and Effects Analysis Prepared by: _____
 Team: _____ Date: _____ (orig)

System / Function	Potential Failure Mode	Potential Effects of the Failure Mode	Severity	Potential Causes of the Failure Mode	Current Preventative Activities	Occurrence	Current Detection Activities	Detection	Priority	Recommended Action(s)	Responsibility & Target Completion Date	Action Results			
												Actions Taken	Severity	Occurrence	Criticality
Drive Train / Move Table	Ballscrew breaks	a) Table collapses b) Table disconnects from motion system c) Mass impacts ground	9	1) Section too small 2) Excessive applied shear force 3) Excessive axial compression causes buckling 4) Excessive deflection due to vibration 5) Decrease in strength due to fatigue	1) Design for applied forces 2) Design controls to run at safe speed 3) Fatigue analysis	2	Observation	8	144	Extensive analysis for static loading and fatigue, additional study into how to decrease severity of part failure	Juan Marco Majano 1/14	Preformed analysis by hand.	5	4	5
Drive Train / Move Table	Ballscrew bends	a) Table doesn't move b) Table binds up c) Mass moves unpredictably d) Deformation to physical components e) Interference with other systems	7	1) Section too small 2) Excessive applied shear force 3) Excessive axial compression causes buckling 4) Excessive deflection due to vibration 5) Decrease in strength due to fatigue	1) Design for applied mass 2) Design controls to run at safe speed 3) Fatigue analysis	3	Observation	6	126	Extensive analysis for static loading and fatigue, additional study into how to decrease severity of part failure	Caleb O'Gorman 1/14	Preformed analysis by hand.	2	2	3
Drive Train / Move Table	Component Misalignment	a) No full range of motion b) Table doesn't move c) Table moves by incorrect amount d) Table binds up e) Interference with other systems	9	1) Loose tolerances during manufacturing 2) Thermal drift 3) Improper installation 4) Bad surface finish	1) Off the shelf parts 2) Installation plan 3) Low tolerance manufacturing for housing	2	Trial runs	3	54	Assemble moving table components on ball screw to run test	Ryan Funchess 5/07	Assembled and run through so far no issues.	5	5	8
Motion Support / support table	Ballnut Breaks	a) Table Collapses b) Table disconnects from motion system. c) Load will not be transferred.	7	1) Section too small 2) Excessive applied shear force	1) Design for applied forces 2) Run at safe speed	2	Observation	5	70	Run through the operation many consecutive times to observe ball nut.	Nick De Simone 5/07	Assembled and run through so far no issues.	7	2	5
Motion Support / support table	Linear Bearings get clogged with debris	a) Table doesn't move b) Table binds up c) Load will not be transferred.	6	1) Dirty rails 2) Contact with external food or drink 3) No lubrication of rails	1) Maintenance plan that involves cleaning and lubrication of rails	5	Observation	2	40	Constant maintenance and assemble lid to help mitigate this.	Lab technician	Lid to be assembled	4	5	2
Motion Support / support table	Rails Break	a) Table Collapses b) Table disconnects from motion system. c) Load will not be transferred.	9	1) Section too small 2) Excessive applied shear force 3) Excessive deflection due to vibration	1) Design for applied forces	2	Observation, maximum load test	5	90	Extensive analysis for static loading and fatigue, additional study into how to decrease severity of part failure, as well as increase detection	Nick De Simone 5/07	Analysis done by hand.	5	2	5
Motion Support / support table	Rails bend	a) Table doesn't move b) Table binds up c) Mass moves unpredictably d) Plastic deformation of physical components e) Interference with other systems	7	1) Section too small 2) Excessive applied shear force 3) Excessive deflection due to vibration	1) Design for applied force	3	Observation	3	54	Run the through the operation and observed the rails in action.	Juan Marco Majano 5/7	Assembled and run through so far no issues.	5	3	3

Design Failure Mode and Effects Analysis

Product: _____

Prepared by: _____

Team: _____

Date: _____ (orig)

System / Function	Potential Failure Mode	Potential Effects of the Failure Mode	Severity	Potential Causes of the Failure Mode	Current Preventative Activities	Current Detection Activities	Detection	Priority	Recommended Action(s)	Responsibility & Target Completion Date	Action Results			
											Actions Taken	Severity	Occurrence	Criticality
Control Motion / Monitor System	Wires come loose/Disconnected	a) Mass Impacts structure with high force b) Mass moves unpredictably c) Mass moves beyond limit d) Electronics Malfunction	4	1) Incorrect soldering 2) Wear on components 3) Outside interference	1) Check wires before starting machine	Observation, check for current flow with multimeter	1	30	Check connection on all wires to make sure they are not loose. If loose wires exist replace them with new ones.	Caleb O'Gorman 5/7	Created housing so that wires won't be interfered with.	2	5	1
Control Motion / Monitor System	Debris in scale housing	a) Mass moves unpredictably b) Electronics Malfunction	6	1) No cleaning action 2) Contact with external food or drink 3) No use of the lid 4) Dirty encoder reader	1) Design of lid for system 2) Development of cleaning plan	Periodic testing of encoder reader	3	36	Constant maintenance of part.	Lab technician	Lid to be assembled	4	4	3
Control Motion / Monitor System	Wires become stuck during reader motion	a) Binding of ballscrew b) Mass moves unpredictably c) Electronics Malfunction d) Destruction of cables e) Destruction of encoder reader	9	1) Extreme motion of table past limits 2) Unexpected cable kinematics	1) Use of cable track 2) Proper space allotted to cable motion	Testing of cable track motion at extreme points	5	90	Look into decreasing severity of failure. This can be mitigated by causing the attachment between wires and encoder reader to fail first, which will cut system power and stop ballscrew rotation. Additional analysis required.	Ryan Funchess 5/14	Ran the table all the way through its travel distance to make sure wires are long enough. Also implemented cable management so there are no snags.	6	2	5
Control Motion / Interact with MCU	Water/outside interference with electronics	a) Electronics malfunction b) Mass moves unpredictably	8	1) User negligence 2) Improper location of system	1) Lid design 2) Insulated electronics 3) Keep system away from hazardous spaces	Restrictions on outside food, responsible user to ensure no misconduct	1	24	Lid will help with this also instructor can request students to not bring liquid around the device.	Instructor	Designed lid to prevent outside interaction	6	3	2
Control Motion / Maintain Safety	Controller Malfunction	a) Mass Impacts structure with high force b) Mass moves unpredictably c) Mass moves beyond limit d) Electronics Malfunction	9	1) Hardware not properly configured 2) Damage of hardware (water spilled, etc)	1) Routine check to ensure functionality 2) Safety protocol to be followed during lab to ensure safety of components	Test limit switch in simulation, push table to extreme point with power off to test kill switch	3	81	Use CPE student to check wiring and set-up is right	Nick De Simone 4/28	Designed a housing for electrical components as well as a safety cover that is connected in series to the power supply	5	2	3
Mass support system/secure payload mass	Screws Break	a) Mass falls off b) Interference with drivetrain and table c) Possible user injury	6	1) Fasteners shear 2) system is dropped 3) Hole shears 4) Separation of joints 5) Excessive preload 6) Threads break	1) Fastener stress analysis	Observation	5	30	Extensive Finite Element Analysis (FEA), as well as specification of the loads that will be acting on the system.	Juan Marco Majano 5/7	Tested system extensively, monitoring for any sign of screws breaking	6	1	3
Mass support system/secure payload mass	screws unscrewed	a) Mass falls off b) Interference with drivetrain and table c) Possible user injury d) System vibration	6	1) Too small of a preload 2) Vibration 3) Excessive applied force	1) Users will ensure screws tight before use 2) System will not be run near natural frequency 3) Design for correct loads	Observation, ensure screws are tight before use	2	36	Add locktight to these fasteners	Ryan Funchess 5/7	Screwed fasteners in tightly	4	2	3

Design Failure Mode and Effects Analysis

Product: _____

Prepared by: _____

Team: _____

Date: _____ (orig)

System / Function	Potential Failure Mode	Potential Effects of the Failure Mode	Severity	Potential Causes of the Failure Mode	Current Preventative Activities	Current Detection Activities	Detection	Priority	Recommended Action(s)	Responsibility & Target Completion Date	Action Results			
											Actions Taken	Severity	Occurrence	Criticality
General/secure components	screws break	a) Interference with other systems b) Parts uncontrollably falling out of system	7	1) Fasteners shear 2) system is dropped 3) Hole shears 4) Separation of joints 5) Excessive preload 6) Threads break	1) Fastener stress analysis	Observation	5	35	Extensive Finite Element Analysis (FEA), as well as specification of the loads that will be acting on the system.	Caleb O'Gorman 5/7	Tested system extensively, monitoring for any sign of screws breaking	6	1	3
General/secure components	screws unscrewed	a) Interference with other systems b) Parts uncontrollably falling out of system	6	1) Too small of a preload 2) Vibration 3) Excessive applied force	1) Users will ensure screws tight before use 2) System will not be run near natural frequency 3) Design for correct loads	Ensure screws are tight before use	2	36	Add locktight to these fasteners	Nick DeSimone 5/7	Screwed fasteners in tightly	4	2	3
Housing / contain system	Housing Breaks	a) hazards are exposed (pinch points, rotating components, etc) b) system components are exposed to damage (dust, user damage, etc) c) user is injured d) Parts fall out uncontrollably	9	1) Section too thin 2) Excessive applied force 3) High stress at stress concentrations 4) Excessive deflection due to vibration 5) System is dropped from a height 6) Lack of support material at critical points	1) Stress analysis for applied forces 2) Drop test 3) Design controls to not run system near natural frequency	Observation	6	108	Extensive Finite Element Analysis (FEA), as well as specification of the loads that will be acting on the system.	JMM 1/14	Designed the housing out of aluminum rather than PLA.	8	0	1
Drive Train/ Move Table	Gearbox Backdrive	a) Damage to components b) Mass moves uncontrollably	6	1) Users turning the gearbox output by hand 2) Output rotates independent of motor	1) Design ballscrew for low moment of inertia compared to motor 2) Rigid coupling between gearbox output and ballscrew 3) Instruct users to not backdrive by hand	Observation, responsible user to ensure no misconduct	2	36	Include GUI feature to prevent this backdrive	Ryan Funchess 5/14	Used a motor powerful enough to not require a gearbox	6	0	0
Control System/Interact with user	UI Not User Friendly	a) Misuse of system b) Inability to use system c) Ineffective as educational model	3	1) Not designing with user in mind 2) Lack of knowledge in designing UI	1) Human Centered Design Approach 2) Test with sample group 3) Recruit CPE Student 4) Research into software engineering	Analyze results of tests with students, receive feedback from professors	1	3	Make different version to see what students will like.	Caleb O'Gorman 5/14	Have showed different designs to different people with experience in MATLAB GUIs and taken their feedback	2	5	2

A.13. Design Verification Plan (DVP&R)

DVP&R - Design Verification Plan (& Report)											
Project:	F12			Sponsor:	Professor Xing						Edit Date:
TEST PLAN										TEST RESULTS	
Test #	Specification	Test Description	Measurements	Acceptance Criteria	Required Facilities/Equipment	Parts Needed	Responsibility	TIMING		Numerical Results	Notes on Testing
								Start date	Finish date		
1	Table Travel Distance	We will manually move the table to the extreme points of its path by manually turning the ballscrew and measure the position of the side of the front of the table.	Length between extreme points of table motion	The table is able to travel the specified distance of 30 cm unimpeded.	Measuring tape	Ballscrew, ballnut, table, linear rails+bearings, thrust and ball bearings + mounts	Juan	2/6/2021	5/19/2021	25 cm	Table only able to travel 25cm due to physical constraints. This translates to <25cm on the GUI side due to overshoot. RECCOMENDATION: Increase system length or decrease width of table.
2	Limit switches must prevent extreme motion of table	Verify connection between computer and sensor and ensure that the controller does not allow motion beyond the speicified limit.	n/a	The table does not move beyond a specified position.	n/a	Entire system	Ryan	TBA	5/19/2021	TRUE	The limit switches effectively prevent further motion. When they are pressed during the run function, they stop the table immediately.
3	Load Requirement	Table + Reaction Supports are able to withstand 5 kg load without deflection and still travel linearly.	n/a	(1) No visible deflection larger than 0.05 in (2) Table travels 30cm under load	Calipers to measure deflection	Frame, Ballscrew, linear rails +bearings, masses	Nick	2/6/2021	5/19/2021	TRUE	No measurable deflection could be measured with a tape measurer.
4	Speed of the Table	Time the table's ability to move between extreme points of motion at maximum speed. Run iseveral times to verify its consistency.	Velocity	Ensure a minimum top speed of 10cm/s.	iPhone timer, measuring tape	Entire system	Juan	TBA	5/19/2021	7.5 cm/s top speed	Under all conditions, we could only get the table to a max speed of 7.5 cm/s. RECCOMENDATION: Get a more powerful motor or higher quality bearings to minimize losses.
5	Mobility of Unit	We will put the system on a scale and record its weight.	System weight	System <= 50 lb	Weight Scale	Entire system	Nick	2/6/2021	5/19/2021	33.2 lb	System was weighed without wooden base using three springs with a known spring constant.
6	Communication between motor and microcontroller	We will send voltages to the motor and ensure that linear speed varies controllably based on Pulse Width Modulation (PWM) Duty Cycle	Voltage, motor output speed	Motor spins at 1200 rpm predictably.	Voltmeter, jumper cables, tape, high speed camera, possibly tachometer	Microcontroller, motor	Nick	2/13/2021	5/19/2021	TRUE	Table speed varies approximately linearly with increasing voltage duty cycle.

DVP&R - Design Verification Plan (& Report)												
Project:	F12			Sponsor:	Professor Xing						Edit Date:	
TEST PLAN											TEST RESULTS	
Test #	Specification	Test Description	Measurements	Acceptance Criteria	Required Facilities/Equipment	Parts Needed	Responsibility	TIMING		Numerical Results	Notes on Testing	
								Start date	Finish date			
7	Communication between rotary encoders and microcontroller	Verify that the encoders yield accurate readings when we turn the shaft a given amount.	Angle of shaft, encoder readings	Observe encoder accuracy within	Voltmeter, protractor, jumper cables	Motor, Microcontroller	Ryan	2/13/2021	5/19/2021	Average Error: 0.29cm	The average was calculated from a series of three runs. The encoder itself is highly accurate, but due to the unprecise method for zeroing the encoder reading, we achieve this bias error. RECCOMENDATION: recalibrate and then decrease the speed of table during zeroing function to avoid overshoot onto limit switch.	
8	Communication between computer and microcontroller	Ensure that the computer can communicate via USB to the microcontroller and run the motor from the computer. Verify that command prompts from the UI will control output pins of the microcontroller.	N/A	Serial communication between the computer and microcontroller is used to run the motor.	Multimeter, USB cable	GUI, Microcontroller, motor	Ryan	TBA	5/19/2021	TRUE	We are able to effectively and accurately control system performance using the designed MATLAB GUI.	
9	User Interface	Have a roommate try the interface to see if they can understand and operate the system effectively.	Time	Roommate is able to use the interface to move the table in under five minutes.	A Roommate, stopwatch	Microcontroller, computer	Caleb	TBA	5/19/2021	Average time: 29.3 s	Test was performed on three different roommates. They had the operation verbally explained to each of them individually and then were asked to input a PID controller and profile for a test run.	
10	Motion System Resolution	Manual turn the ball screw 30 times and measure the distance it traveled. (ballscrew pitch)	Distance	The table should move 15 cm.	Caliper	Ballscrew, ballnut, table	Caleb	2/6/2021	5/19/2021	15.00 cm on 2 trials	The manufacturers listed ballscrew pitch was, unsurprisingly, highly accurate.	

A.14. Design Selection Analysis

Motor Sizing

Motor Sizing		
<u>Quantity</u>	<u>Value</u>	<u>Unit</u>
Payload Mass	25	kg
Min Speed	0.1	m/s
Ballscrew Efficiency	0.8	
Motor Projected Efficiency	0.75	
Mean Diameter	10	mm
Pitch	5	mm
Lead	5	mm
Friction Coefficient	0.74	
Friction Force	181.3	N
Torque to Drive	0.18	Nm
Angular Velocity Required	20	rot/s
Angular Velocity Required	1200	rpm
Angular Velocity Required	125.66	rad/s
Motor Power	30.22	W

Ballscrew Critical Speed, Buckling, Static Stress Analysis

Material	Pn E Gpa	Y (lb/m^3)	g (m/s^2)	Dens (kg/m^3)	Sy (Pa)	Su (pa)	Geom	r (m)	d (m)	A (m^2)	Lead (m/rev)	I (m^4)	L (m)
1040 Steel	1.90E+11	76959.45	9.81	7845	4.15E+08	6.20E+08		0.0125	0.025	0.078539816	0.005	1.917E-08	0.35

Critical Speed			
Wcr	1.96E+02 rad/s	1870.835 rpm	
Ncr	1.559028826		
Run Speed			
Linear (m/s)	0.1	Angular (rpm)	1200
		Angular (rad/s)	125.6637

Buckling - Johnson	
Pcr	7.34E+04 N
Nbuck	4.03E+02

Input	
Intermediate Value	
Key Value	

Bending Moment	Payload W (N)	Max M (N-m)	σ (N/m^2)
	180	15.75	10267404
Thread Stress	X (N/m^2)	Y (N/m^2)	τ (N/m^2)
	2780755.166	-370767.36	58670.878
Total Stresses	X	Y	τ (N/m^2)
	2780755.166	10267403.7	58670.878 (MPa)
Principal Stresses			
	2.780755166	10.2674037	0.0586709
Tresca Failure Theory	c	6.52407943 MPa	
	r	3.74378402	
	sig1	10.2678634	
	sig2	2.78029541	
	sig3	0	
Tresca Failure Theory	FS	4.04E+01	

Ballscrew Fatigue Analysis

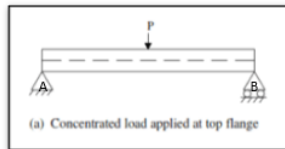
Endurance Limit			
Sut	400 Mpa	Assumed 1020 steel	
d	18.91499138 mm		
Sy	415 Mpa		
S'e	200 Mpa		
ka	0.921787138	See table 6-2 at right	
kb	0.905322736	diameter correction	
kc	1	Leave, don't use with von mises	
kd	1	temp	
ke	0.814	Reliability, Table 6-5 at right 99% Reliability	
Kt	1	stress conc table	
q	1	notch sensitivity	
Kts	1		
Kf	1		
Kfs	1		
Se	135.8590183 mpa		
Bending Moment(alt)	18 Nm		
Torque(const)	0.18 Nm		
SigABend	27092817.24 pa		
SigMBend	0 pa	usually 0	
SigATors	0 pa	usually 0	
SigMTorsi	135464.0862 pa		
SigAVonMis	27092817.24 pa		
SigMVonMis	234630.6799 pa		
Mod Good FS	4.999870215	Factor of Safety	
Yielding	15.18619672	Factor of safety	

Preliminary Rail Sizing

Purpose: This sheet calculates the minimum radius of the linear railway based on given load and shaft material.

Model:

- Beam:** Supported at Both Ends with One-Fixed and One-Roller Support.
- Load:** Point Load at center of beam (area of max deflection). Note the actual loading is a distributed weight, but the point load is conservative analysis.



Deflection Analysis:

Givens:	Value	Units
Load	50	lb
(User Enters Values) Material	Carbon Steel	(-)
Yield Strength	490	Mpa
Shaft Length	50	cm

Calculations:

		Value	Units
FBD	Force "P"	222.45	N
	Reaction "A"	111.22	N
	Reaction "B"	111.22	N

		Value	Units
Shear & Moment	Max Shear "V"	111.22	N
	Max Moment "M"	27.81	N-m

Sizing:

	Value	Units
Shaft	6.01	mm

*minimum railway shaft radius. Choose a linear guide shaft that is greater than or equal to this value

Bearing Analysis

C10 (90% reliability)	Find Below	
Manufacturer Rating Num Cycles	90000000	Lr
$L(Laplace L)$	3000	Hours for application
n	1200	rpm
Ld	216000000	Revolutions (use 1 mill usually, unless timken)\
a	3	a=10/3 cyl, 3 ball
Fd	50	lbf(converted from kN)
C10=Fr (Lb)	66.943295	lbf
C10=Fr (SI)	0.2977785	kN

A.15. Drawing Package

10000 – Top Level Assembly

11000 - Drive Train Assembly

11100 – Motor Assembly

11110 – Motor

11120 - Screws

11200 - Ballscrew Assembly

11210 – Ball Bearing Assembly

11211 – WBF-20 End Support

11212 – Screws

11220 – Thrust Bearing Assembly

11221 – Thrust Bearing Drawing

11222 – Screws

11230 – Ballscrew Data Sheet

11300 – Ballnut Assembly

11310 – Ballnut Data Sheet

11320 – Ballnut Bracket Drawing

11330 – M6 X 0.8 Screws Drawing

11400 - Flexible Shaft Coupling Assembly

11410 - 11/16" Flexible Shaft Coupling Iron Hub Drawing

11420 - 1/4" Flexible Shaft Coupling Iron Hub Drawing

11430 - Buna-N Rubber Spider Drawing

12000 - Rail Assembly

12100 – Rails Modification Drawing

12200 – Screws

13000 - Mass Support Assembly

13110 - Table Drawing

13210 - Plate Mass

13220 – Bolts Drawing

13230 – Nuts Drawing

14000 – Structure Assembly

14100 – Base Plate Drawing

14200 – Rail Spacer Drawing

14300 – Datum Face Drawing

14400 – Motor Face Drawing

14500 – Ball Bearing Face Drawing

14600 – Motor-Side Leg Drawing

14700 – Opposite-Side Leg Drawing

14800 – Dowel Pin Data Sheet

14900 – ¼” in – 20 Bolt Drawing

14110 – ¼” in – 20 Nut Drawing

14120 – Metal Bracket Data Sheet

14130 – Plastic Corner Bracket Data Sheet

14140 – 8-32 Screws Drawing

14150 – 8-32 Nuts Drawing

14260 - Lid Drawing

15000 – Microcontroller Wiring Diagram

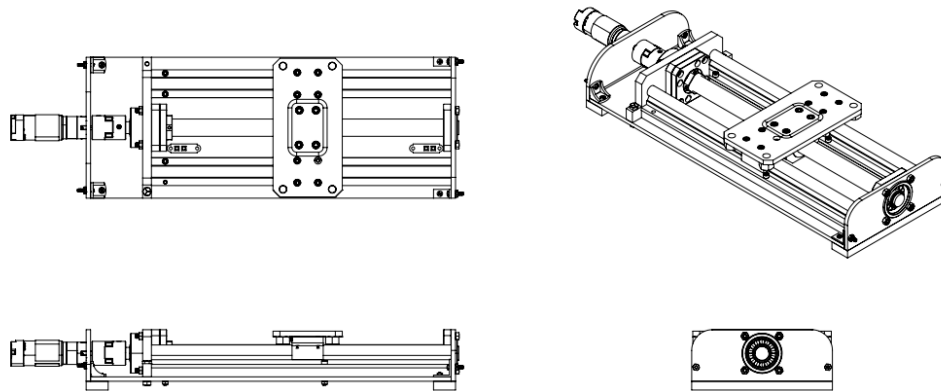
15100 - Microcontroller Data Sheet

15300 - Limit Switch Drawing

15400 – Breadboard Data Sheet

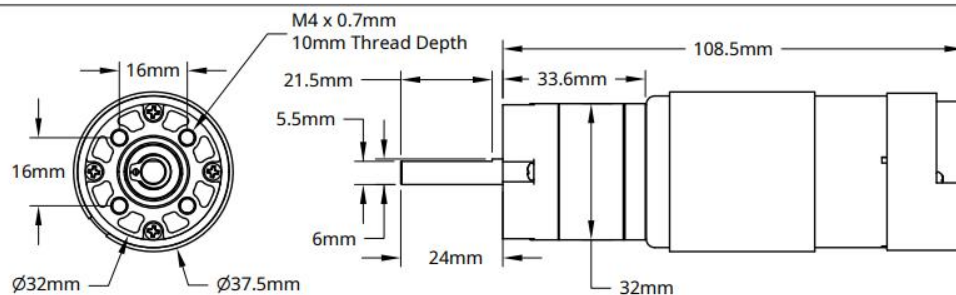
15500 – Wire Data Sheet

16000 – Lid Data Sheet



Cal Poly Mechanical Engineering ME 430 - Spring 2021	CNC FEED DRIVE pt #: 10000	FDR MODEL Not Asd: ASSEMBLY	Title: FEED DRIVE ASSEMBLY Date: 6/3/21	Scale: 1=4	Drwn. By: RYAN FUNCHES Chkd. By: F12
---	-------------------------------	--------------------------------	--	------------	---

SOLIDWORKS Educational Product. For Instructional Use Only.



SKU: 5202-0002-0003
 Gearbox Style: Planetary
 No-Load Speed @ 12VDC: 1,620 RPM
 No-Load Current @ 12VDC: 0.25A
 Stall Torque @ 12VDC: 5.4 kg.cm (75.8 oz-in)
 Stall Current @ 12VDC: 9.2A
 Gear Ratio: 3.7:1
 Gear Material:
 Casing: Steel Ring Gear
 Stage 1: Steel Pinion Gear, Steel Orbital Gears
 Output Shaft Support: Dual Ball Bearing
 Motor Size: RS-555
 Weight: 387g
 Wire Length: 470mm with 3.5mm Male Bullet Connectors
 Encoder Shaft: 28 PPR (7 rises of channel A)
 Gearbox Output Shaft: 103.6 PPR (25.9 rises of channel A)



Ball Screws and Related Products.....Technical Catalog



Table of Contents



Ball screws & nuts

Expanded stock sizes now include
16mm, 20mm, 25mm & 32mm
diameters in 5mm & 10mm leads

Page 1...General info/sizes
Page 2...Ball nut dimensions
and load ratings
Page 3...End machining specs



BK & BF supports

Pillow block style

Page 4...BK fixed supports
Page 5...BF simple/floating supports



FK & FF supports

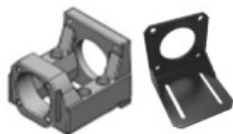
Flange style

Page 6...FK fixed supports
Page 7...FF simple/floating supports



Ball nut brackets

Page 8



Motor brackets

Pages 9 & 10



Flexible couplings

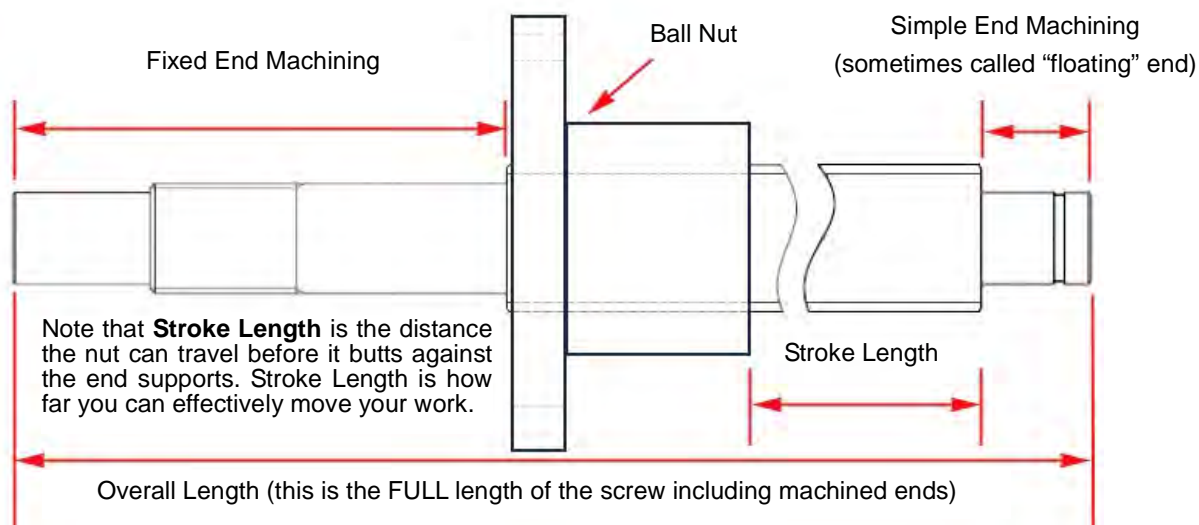
(connect motor to ball screw)

Pages 11 & 12

Precision Rolled Ball Screws



Need C5 rolled or ground screws? Contact us for a quick quote.



Typical part number: **BSFU1605-0350-FS**

BSFU16 = Our ball nut identifier and diameter (in mm)

05 = Lead for screw in mm (one turn moves nut this many mm)

0350 = Overall length of screw in mm (end to end)

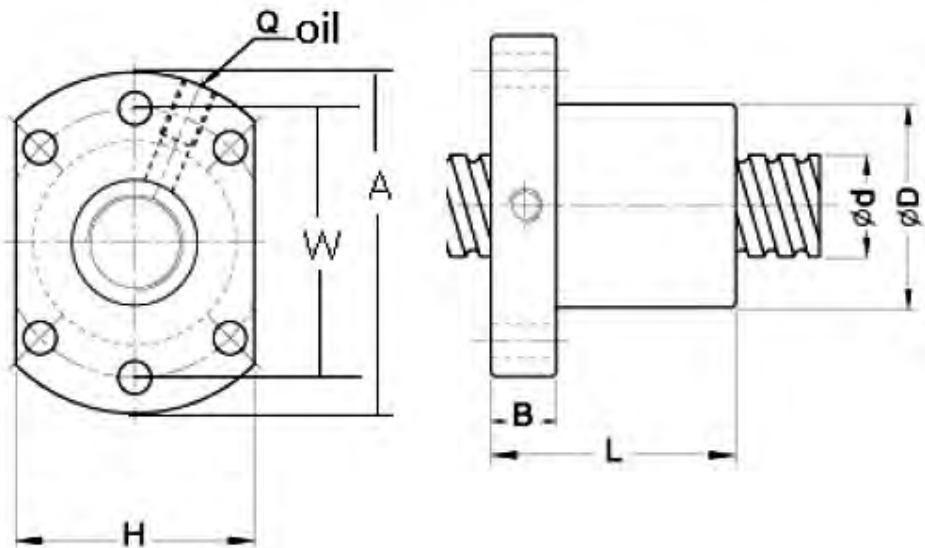
FS = Fixed & Simple end machining

Our stock program includes these parts/sizes

BSFU1605-0350-FS	BSFU1610-0350-FS	BSFU2005-0500-FS	BSFU2505-0500-FS	BSFU3205-0500-FS
BSFU1605-0450-FS	BSFU1610-0450-FS	BSFU2005-0750-FS	BSFU2505-0750-FS	BSFU3205-0750-FS
BSFU1605-0550-FS	BSFU1610-0550-FS	BSFU2005-1000-FS	BSFU2505-1000-FS	BSFU3205-1000-FS
BSFU1605-0650-FS	BSFU1610-0650-FS	BSFU2005-1500-FS	BSFU2505-1500-FS	BSFU3205-1500-FS
BSFU1605-0750-FS	BSFU1610-0750-FS	BSFU2005-2000-FS	BSFU2505-2000-FS	BSFU3205-2000-FS
BSFU1605-0850-FS	BSFU1610-0850-FS	BSFU2010-0500-FS	BSFU2510-0500-FS	BSFU3210-0500-FS
BSFU1605-0950-FS	BSFU1610-0950-FS	BSFU2010-0750-FS	BSFU2510-0750-FS	BSFU3210-0750-FS
BSFU1605-1150-FS	BSFU1610-1150-FS	BSFU2010-1000-FS	BSFU2510-1000-FS	BSFU3210-1000-FS
BSFU1605-1350-FS	BSFU1610-1350-FS	BSFU2010-1500-FS	BSFU2510-1500-FS	BSFU3210-1500-FS
BSFU1605-1550-FS	BSFU1610-1550-FS	BSFU2010-2000-FS	BSFU2510-2000-FS	BSFU3210-2000-FS

Contact us for custom screws of any size, length, lead & material grade!

Ballnut/Ball Screw Tech Data



Note: Ball nuts are supplied with non-removable flange as shown. Other styles available.

Material Type	Dia. d	Lead	Ball Dia.	D	A	B	L	W	X	H	Oil Q	# of Circuits	Dyn. kgf Ca	Sta. kgf Coa	Rigidity Kgf/um K
BSFU1605	16	5	3.175	28	48	10	50	38	5.5	40	M6	4	1380	3052	32
BSFU1610	16	10	3.175	28	48	10	44	38	5.5	40	M6	3	1103	2401	26
BSFU2005	20	5	3.175	36	58	10	51	47	6.6	44	M6	4	1551	3875	39
BSFU2010	20	10	3.175	36	58	10	44	47	6.6	44	M6	3	1516	3833	21
BSFU2505	25	5	3.175	40	62	10	51	51	6.6	48	M6	4	1724	4904	45
BSFU2510	25	10	4.762	40	62	12	85	51	6.6	48	M6	4	2954	7295	50
BSFU3205	32	5	3.175	50	80	12	52	65	9	62	M6	4	1922	6343	54
BSFU3210	32	10	6.35	50	80	12	90	65	9	62	M6	4	4805	12208	61

We stock various lengths in ALL sizes shown above. Refer to page 1 of this document for list of stocked sizes. Special sizes also available in these materials and many others with 2-3 week lead time. Rolled screws. Ground screws. Just about any diameter and lead you could need. Just ask us for a quick, no-obligation quote.

Rolled screws have Class 7 accuracy rating.

ACCURACY

Precision Rolled Screw Runout: +/- 0.050/300mm or 0.002"/12".

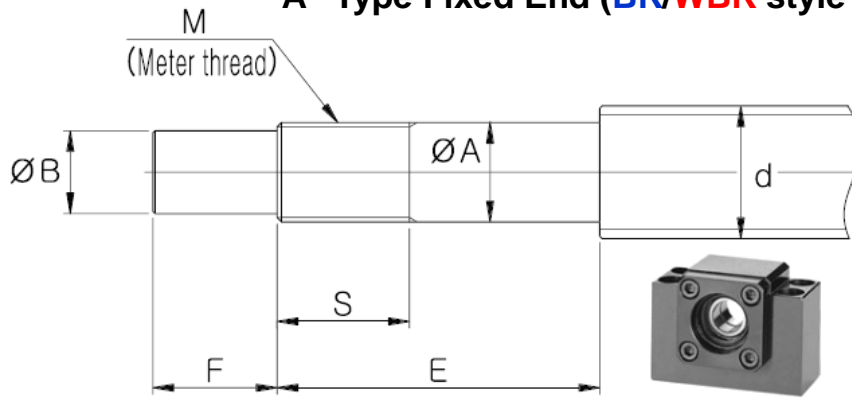
Ballnut: Single ballnut with anti-backlash defined as <0.015mm or <0.0006".

Note: Our standard ball screws and most of our custom screws are supplied with the "BSFU" style ball nut. This is a very common design of ball nut which also carries this designation: Type DIN 69051. The ball nuts we provide are individually loaded and matched to screws for longer life and better performance.

With other sizes/leads, other ball nut styles may be utilized. We will advise you about the type of ball nut available for a given size at the time of quote.

Typical/Recommended End Machining

“A” Type Fixed End (BK/WBK style mount)

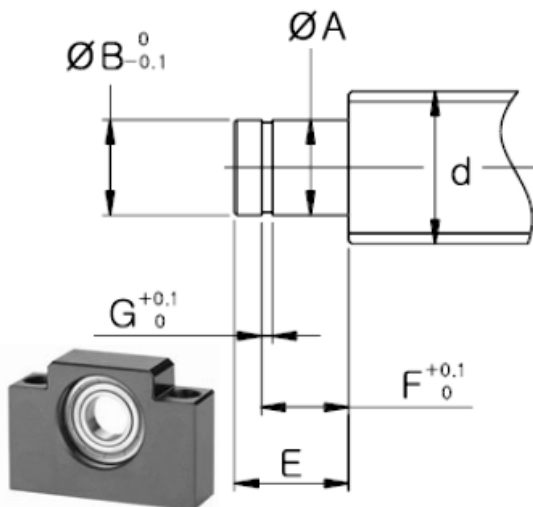


Important note: In some cases slightly different machining is required on the fixed end depending on whether you are using a BK mount or WBK mount. Before we can provide a custom screw, we MUST know how you intend to support the ball screw.

All dimensions in MM

Support PN	Screw Ø d	Bearing Ø A	Motor Coupling B	E	F	M	S
BK10/FK10*	12/14/15	10	8	36	15	M10 x 1	16
BK12/FK12*	14/15/16	12	10	36	15	M12 x 1	14
BK15	18/20	15	12	40	20	M15 x 1	12
FK15*	18/20	15	12	47	20	M15 x 1	12
BK20	25/28	20	17	53	25	M20 x 1	15
F20*	25/28	20	17	62	25	M20 x 1	15
BK25	32/36	25	20	65	30	M25 x 1.5	18
FK25*	32/36	25	20	76	30	M25 x 1.5	18
BK30/FK30*	36/40	30	25	72	38	M30 x 1.5	25
BK35	45	35	30	81	45	M35 x 1.5	28
BK40	50	40	35	93	50	M40 x 1.5	35

*Note that some manufacturers refer to their flange type fixed end supports as “FK” style and others call them “WBK” style. Our provider uses the FK designation.



“B” Type Simple End (for BF & FF style mount)

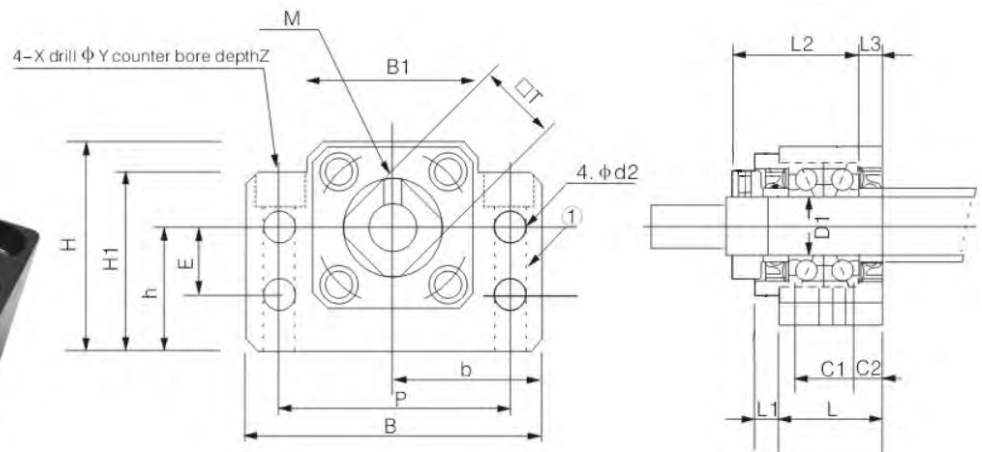
Support PN	Screw Ø d	Bearing Ø A	E	Snap Ring Flute		
				B	F	G
BF10/FF10*	12/14/15	8	10	7.6	7.9	0.9
BF12/FF12*	14/15/16	10	11	9.6	9.15	1.15
BF15/FF15*	18/20	15	13	14.3	10.15	1.15
BF20/FF20*	25 thru 30	20	19	19	15.35	1.35
BF25/FF25*	28 thru 36	25	20	23.9	16.35	1.35
BF30/FF30*	36/40	30	21	28.6	17.75	1.75
BF35	40/45	35	22	33	18.75	1.75
BF40	50	40	23	38	19.75	1.95

While BK and BF end supports are shown in this illustration, the same machining works with WBK/WBF.

*Note that some manufacturers refer to their flange type simple end supports as “FF” style and others call them “WBF” style. Our provider uses the FF designation.

BK

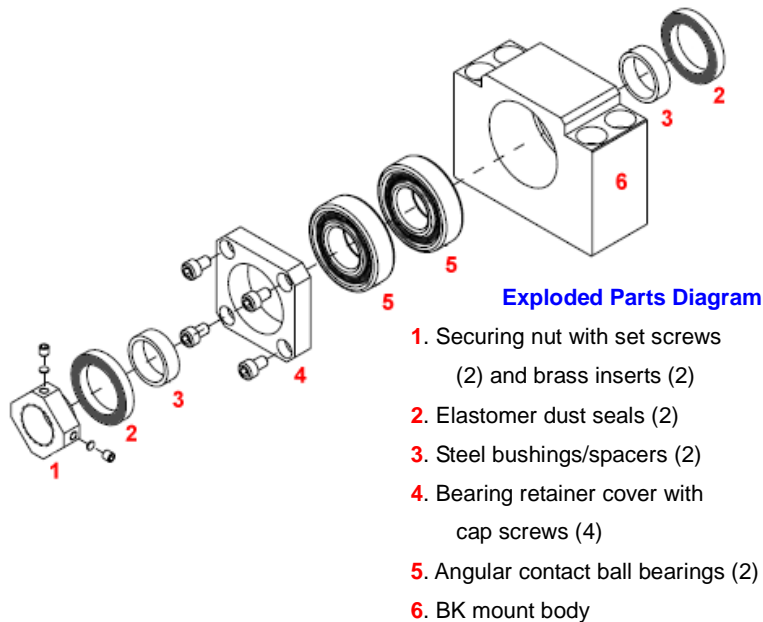
Fixed Side



Dimensions in MM

Model	D1	L	L1	L2	L3	B	B1	H	H1	b	E	h	P	C1	C2	d2	X	Y	Z	T	M
BK10	10	25	6	29.5	5	60	34	39	32.5	30	15	22	46	13	6	5.5	6.3	10.5	5	16	M3
BK12	12	25	6	29.5	5	60	34	42	32.5	30	18	25	46	13	6	5.5	6.3	10.5	5.5	19	M3
BK15	15	27	6	32	6	70	38	47	38	35	18	28	54	15	6	5.5	6.3	10.5	6.5	22	M3
BK17	17	35	10	44	7	86	48	63	55	43	28	39	68	19	8	6.6	8.7	14	8.6	24	M4
BK20	20	35	6	43	8	88	50	59	50	44	22	34	70	19	8	6.6	8.7	14	8.5	30	M4
BK25	25	42	6	54	9	106	62	79	70	53	33	48	85	22	10	9	10.7	17.5	10.5	35	M5
BK30	30	45	6	61	9	128	74	88	78	64	33	51	102	23	11	11	13.7	20	13	40	M6
BK35	35	50	10	67	12	140	86	95	79	70	35	52	114	26	12	11	13.7	20	13	50	M8
BK40	40	61	10	76	15	160	98	109	90	80	37	60	130	33	14	14	17.7	26	17.5	50	M8

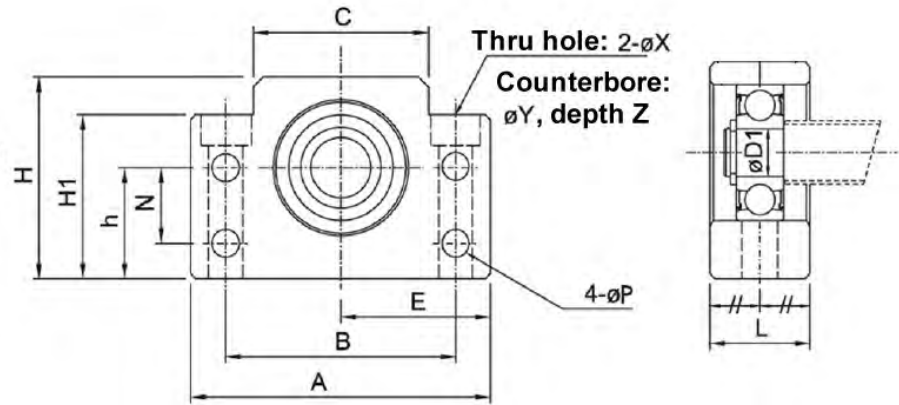
Note: BK mounts come complete with bearing spacer bushings and locking nut. Locking nut may be supplied with separate brass inserts or other design to protect threads when set screws are tightened. See exploded diagram below for typical components.



Load Ratings/Speed Information

Model	Static Load (kgf)	Dynamic Load (kgf)	Max Speed (rpm)
BK10	266	133	16,800
BK12	305	153	15,400
BK15	350	175	13,300
BK17	610	305	11,200
BK20	670	335	10,500
BK25	1,050	525	8,400
BK30	1,510	755	7,000
BK35	1,870	1,202	4,200
BK40	2,340	1,504	3,710

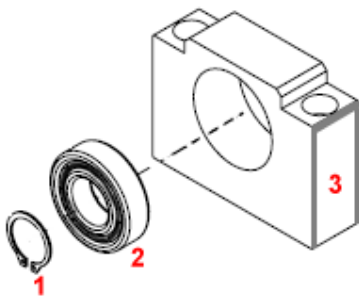
Note that set screws/inserts included with fixed supports are sometimes supplied as a single unit where a brass protective tip is integrated into the set screw or securing nut is of special design to protect threads.

BF**Simple/floating support**

Dimensions in MM

Model	$\phi D1$	A	L	B	C	H	H1	E	X	Y	Z	N	h	P	Snap Ring
BF10	8	60	20	46	34	39	32.5	30	6.3	10.8	5	15	22	5.5	S08
BF12	10	60	20	46	34	43	32.5	30	6.3	10.8	5.5	18	25	5.5	S10
BF15	15	70	20	54	40	48	38	35	6.3	11	6.5	18	28	5.5	S15
BF17	17	86	23	68	50	64	55	43	8.7	14	8.6	28	39	6.6	S17
BF20	20	88	26	70	52	60	50	44	8.7	14	8.6	22	34	6.6	S20
BF25	25	106	30	85	64	80	70	53	10.7	10.7	11	33	48	9	S25
BF30	30	128	32	102	76	89	78	64	13.7	13.7	13	33	51	11	S30
BF35	35	140	32	114	88	96	79	70	13.7	13.7	13	35	52	11	S35
BF40	40	160	37	130	100	110	90	80	17.7	17.7	17.7	37	60	14	S40

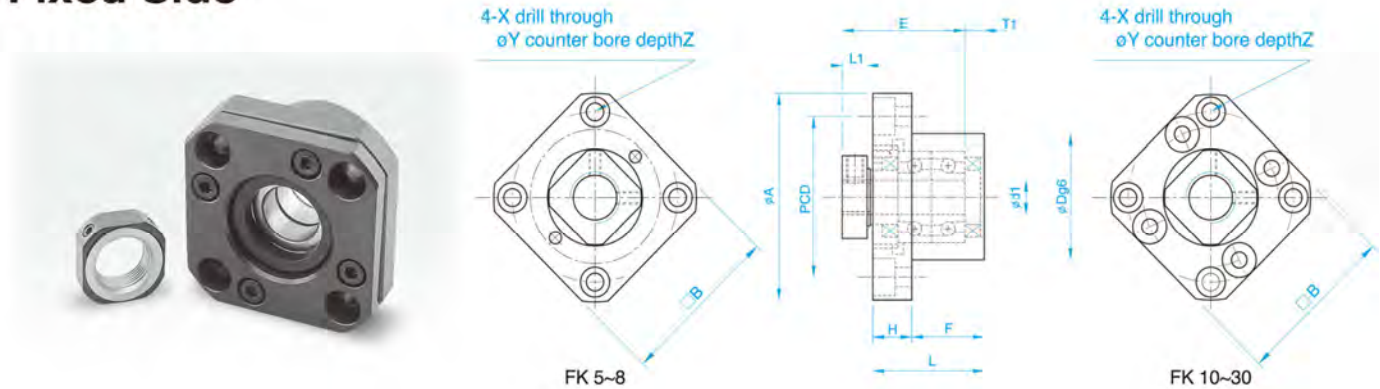
Note: BF mounts are supplied with circlip to secure ball screw to bearing. Ball bearing may be packaged separately in sealed package, but it simply slides into the housing without the need for tools or special skills.

**Exploded Parts Diagram**

1. Circlip/snap ring retaining fastener
2. Sealed ball bearing
3. BF mount body

FK

Fixed Side

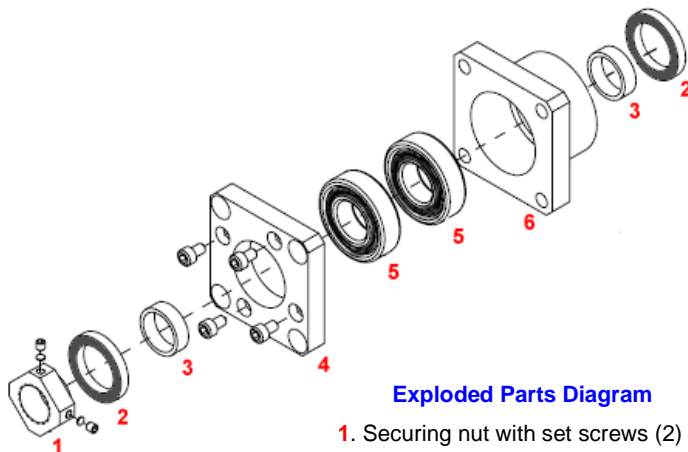


Dimensions in MM

Model	d1	L	H	F	E	Dg6	A	PCD	B	L1	T1	X	Y	Z
FK5	5	16.5	6	10.5	18.5	20	34	26	26	5.5	3.5	3.4	6.5	4
FK6	6	20	7	13	22	22	36	28	28	5.5	3.5	3.4	6.5	4
FK8	8	23	9	14	26	28	43	35	35	7	4	3.4	6.5	4
FK10	10	27	10	17	29.5	34	52	42	42	7.5	5	4.5	8	4
FK12	12	27	10	17	29.5	36	54	44	44	7.5	5	4.5	8	4
FK15	15	34	17	17	36	40	63	50	52	10	6	5.5	9.5	6
FK20	20	52	22	30	50	57	85	70	68	8	10	6.6	11	10
FK25	25	57	27	30	60	63	98	80	79	13	10	9	15	13
FK30	30	62	30	32	61	75	117	95	93	11	12	11	17.5	15

Yellow shaded area above indicates sizes we do not stock but can obtain as special order in QUANTITY

Notes: FK mounts come complete with bearing spacer bushings, locking nut and set screws for locking nut. Some manufacturers refer to their flange type simple end supports as “FK” style and others call them “WBK” style. Our present provider uses the FK designation.



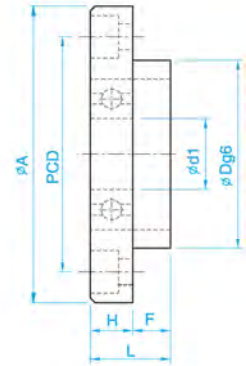
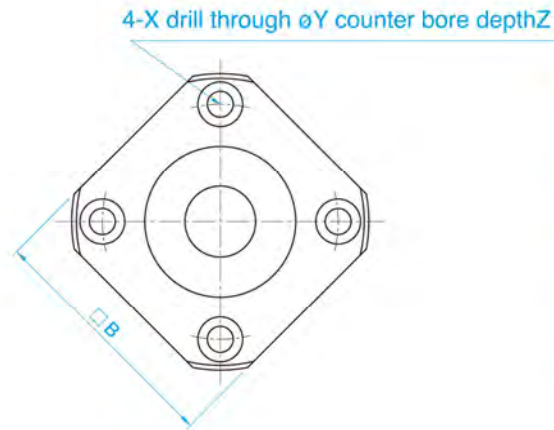
Exploded Parts Diagram

1. Securing nut with set screws (2) and brass inserts (2)
2. Elastomer dust seals (2)
3. Steel bushings/spacers (2)
4. Bearing retainer cover with cap screws (4)
5. Angular contact ball bearings (2)
6. FK mount body

Load Ratings/Speed Information

Model	Static Load (kgf)	Dynamic Load (kgf)	Max Speed (rpm)
FK10	266	133	16,800
FK 12	305	153	15,400
FK 15	350	175	13,300
FK 17	610	305	11,200
FK 20	845	423	9,300
FK 25	1,050	525	8,400
FK 30	1,510	755	7,000

Note that set screws/inserts included with fixed supports are sometimes supplied as a single unit where a brass protective tip is actually attached to the set screw.

FF**Floating Side**

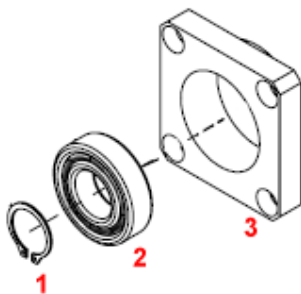
Dimensions in MM

Model	d1	L	H	F	Dg6	A	PCD	B	X	Y	Z
FF6	6	10	6	4	22	36	28	28	3.4	6.5	4
FF10	8	12	7	5	34	43	35	35	4.5	6.5	4
FF12	10	15	7	8	36	52	42	42	4.5	8	4
FF15	15	17	9	8	40	63	50	52	5.5	9.5	5.5
FF20	20	20	11	9	57	85	70	68	6.6	11	6.5
FF25	25	24	14	10	63	98	80	79	9	14	8.5
FF30	30	27	18	9	75	117	95	93	11	17.5	11

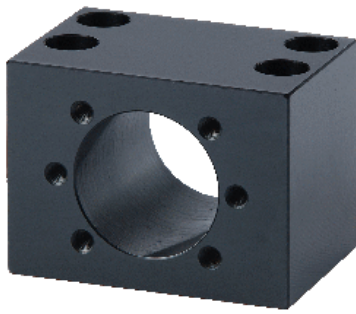
Yellow shaded area above indicates size we do not stock but can obtain as special order in QUANTITY

Some manufacturers refer to this type of end support with the "FF" designation while others use "WBF." They are, in fact, the same in terms of size and function.

Note: FF mounts are supplied with circlip to secure ball screw to bearing. Sealed ball bearing may be packaged separately, but it simply slides into the support body without need for tools or special skills.

**Exploded Parts Diagram**

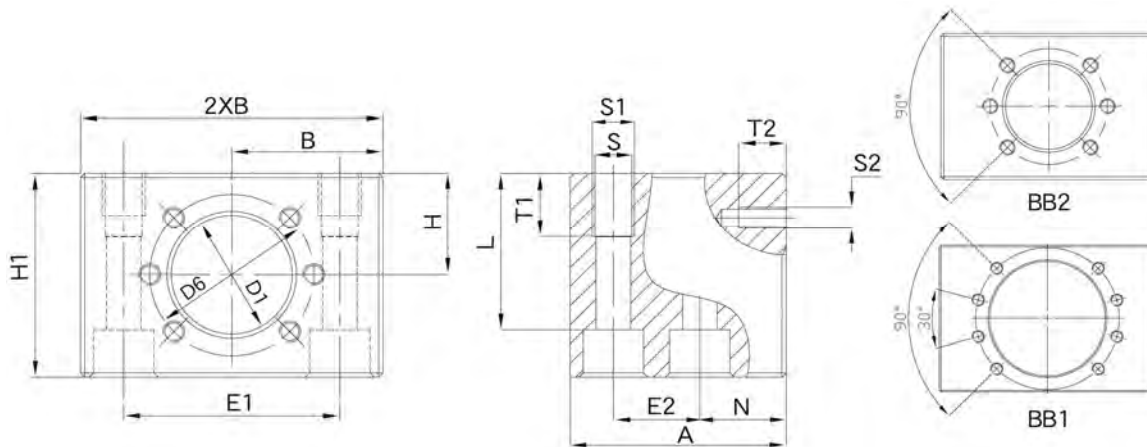
1. Circlip/snap ring retaining fastener
2. Sealed ball bearing
3. FF mount body



MGD Style Ball Screw Nut Bracket

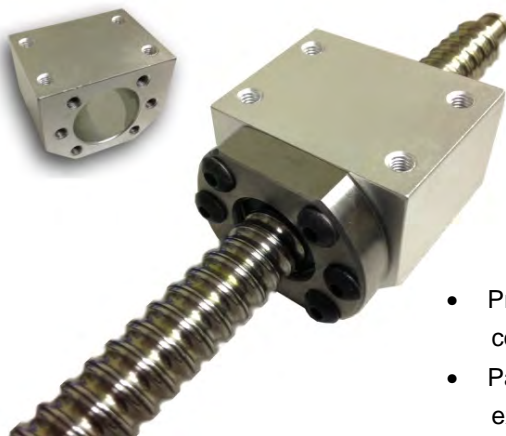
(machined from steel)

We stock the MGD16, MGD20 and MGD25 sizes. Other sizes can be special-ordered. CAD drawings are available from our website.



Unit : mm

Model No.	Size d ₀ ×P	D1±0.1	D6	A	B±0.1	H±0.1	H1	E1	E2	N	S	S1	T1	S2	T2	ISO4762	L	Weight (Gms)
MGD 16	16×5 16×16	28.4	38	50	35	24	48	50±0.1	20±0.1	20	8.4	M10	15	M5	10	BB2	M8	37 0.91
MGD 20	20×5 20×20	36.4	47	55	37.5	28	56	55±0.1	23±0.1	22	8.4	M10	15	M6	11	BB2	M8	45 1.18
MGD 25	25×5 25×25	40.4	51	55	40	30	60	60±0.1	23±0.1	22	8.4	M10	15	M6	11	BB2	M8	49 1.33
MGD32S	32×5 32×10	50.4	65	70	45	35	70	70±0.1	45±0.1	12.5		M12					M12	- 2.5
MGD 32	32×20 32×32				50			75±0.1	30±0.1	27		M16	20	M8	14	BB2	M12	52 2.77
MGD 40	40×5 40×12 40×20 40×40	63.4	78	80	60	42	84	90±0.1	35±0.1	31	15	M18	25	M8	17	BB1	M14	66 3.61



HD Style Ball Screw Nut Bracket

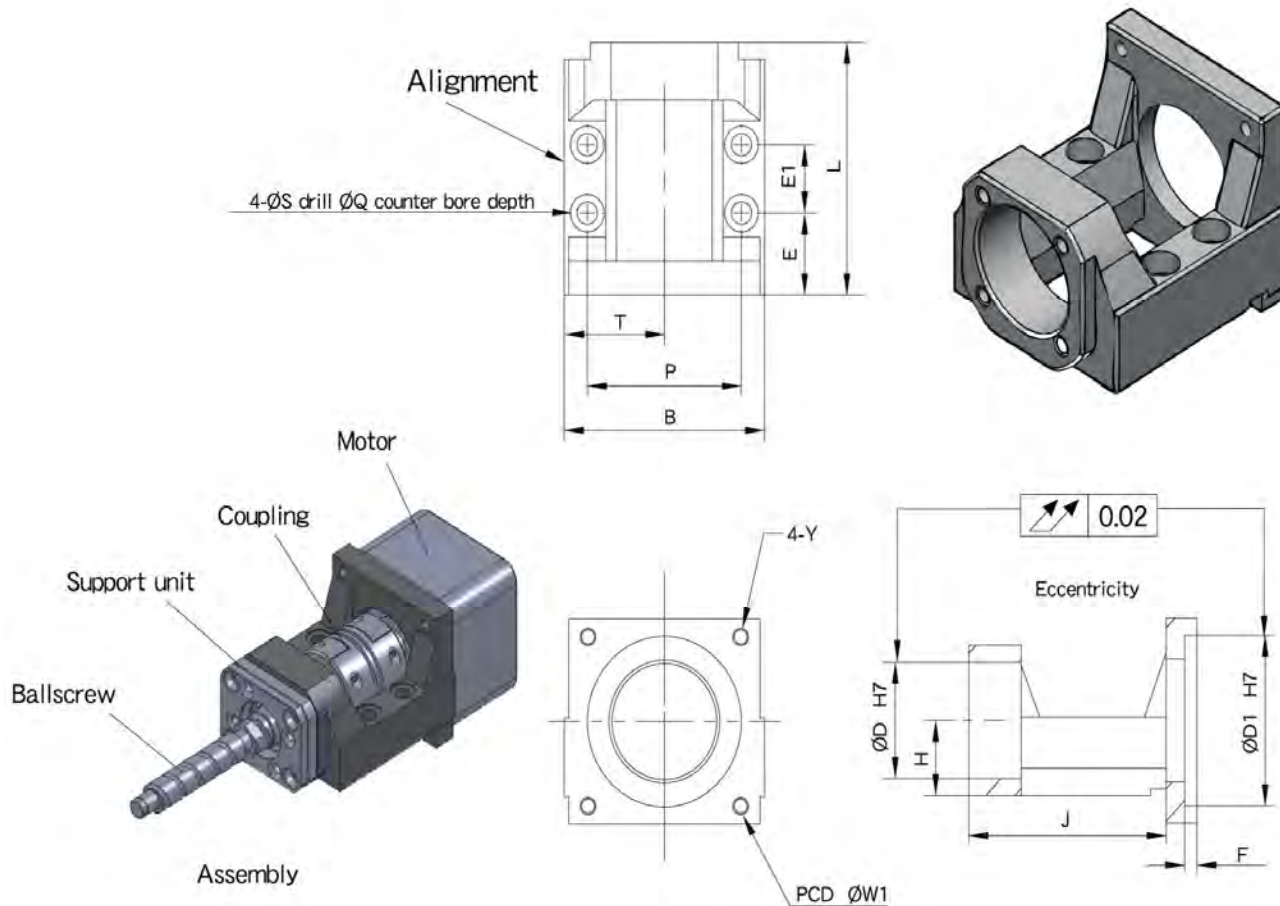
(machined from aluminum)

We stock the HD16, HD20, HD25 and HD32 sizes. CAD drawings and PDF spec sheets are available from our website.

- Precision machined from aluminum. The HD brackets are excellent quality but cost significantly less than MGD brackets above.
- Part numbers indicate the size screw upon which these are used. For example, the HD16 model is used in conjunction with the ball nut typically installed on a 16mm Ø screw. The barrel of the ball nut for this size is usually 28mm Ø, so the receiving bore in the bracket is just in excess of that.

MBA Style Motor Brackets

These brackets make it easy to build a ball screw assembly by combining common motors, flexible motor couplings and ball screw end supports. Save both time and money...and get a precision result FAST.



Model	D1	W1	Y	D	L	H ±0.02	B	P	T	S	Q	E	E1	F	J	Wt. (kgs)	Fixed Side	Floating Side
MBA12-C (NEMA 23)	38.1	66.7	M4	36	74	25	65	50	32.5	6.6	11	24	20	5	65	0.71	FK12 WBK12	BF12
MBA12-D	50	70	M5	36	74	25	65	50	32.5	6.6	11	24	20	5	65	0.71	FK12 WBK12	BF12
MBA15-C (NEMA 23)	38.1	66.7	M4	40	82	28	70	55	35	6.6	11	24	28	5	73	1.4	FK15 WBK15	BF15
MBA-15-D	50	70	M5	40	84	28	70	55	35	6.6	11	25	28	5	74	1.4	FK15 WBK15	BF15
MBA-15-E	70	90	M6	40	94	28	88	70	44	8.5	14	30	28	6	82	1.4	FK15 WBK15	BF15
MBA-15-F (NEMA 34)	73	98.4	M6	40	92	28	88	70	44	8.5	14	29	28	6	81	1.4	FK15 WBK15	BF15
MBA20-D	50	70	M5	57	113	34	88	70	44	8.5	14	29	42	5	-	1.61	FKA20	BF20
MBA20-E	70	90	M6	57	113	34	88	70	44	8.5	14	29	42	6	102	1.61	FKA20	BF20
MBA20-F (NEMA 34)	73	98.4	M6	57	113	34	88	70	44	8.5	14	29	42	6	102	1.61	FKA20	BF20

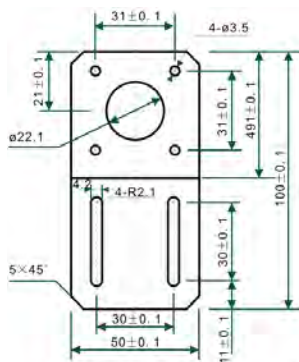
NEMA Motor Brackets



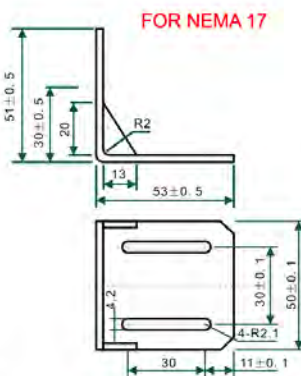
These sturdy steel brackets make it easy to mount NEMA standard frame size motors. We stock NEMA 17, NEMA 23 and NEMA 34 versions. Just bolt up your motor, and you are ready to attach it to a ball screw or other device.

Quality construction with durable black paint finish. Welded corner gussets provide extra rigidity. **VALUE PRICED!**

See images below for dimensions or visit our website to download full-size PDF spec sheets.

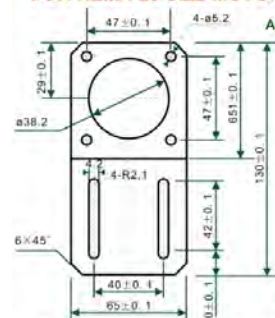


Part Number: NEMA17-MB

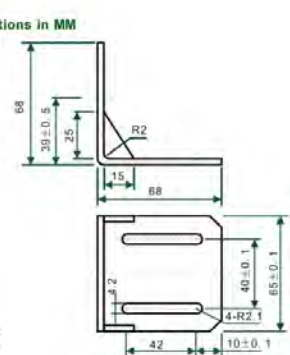


FOR NEMA 17

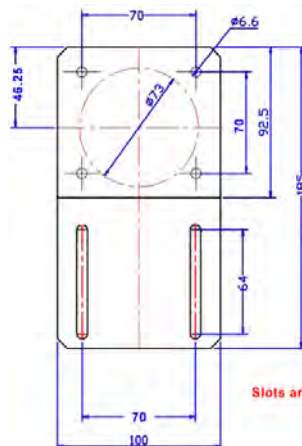
FOR NEMA 23 SIZE MOTORS



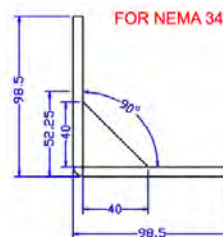
Part Number: NEMA23-MB



All dimensions in MM



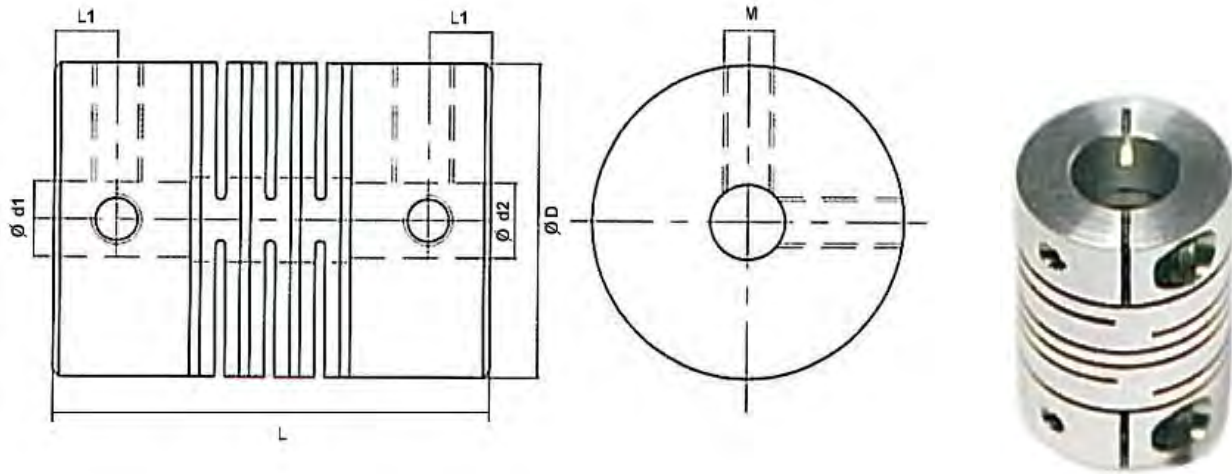
Part Number: NEMA34-MB



FOR NEMA 34

Slots are 6.5mm

DR1-C Flexible Motor Couplings (stainless steel construction)



The DR1-C flexible motor coupling line provides an economical way to connect motors to ball screws and other shafts via clamping pressure applied by two set screws. **Durable stainless steel construction.** These couplings offer higher torque ratings than similar couplings manufactured from aluminum.

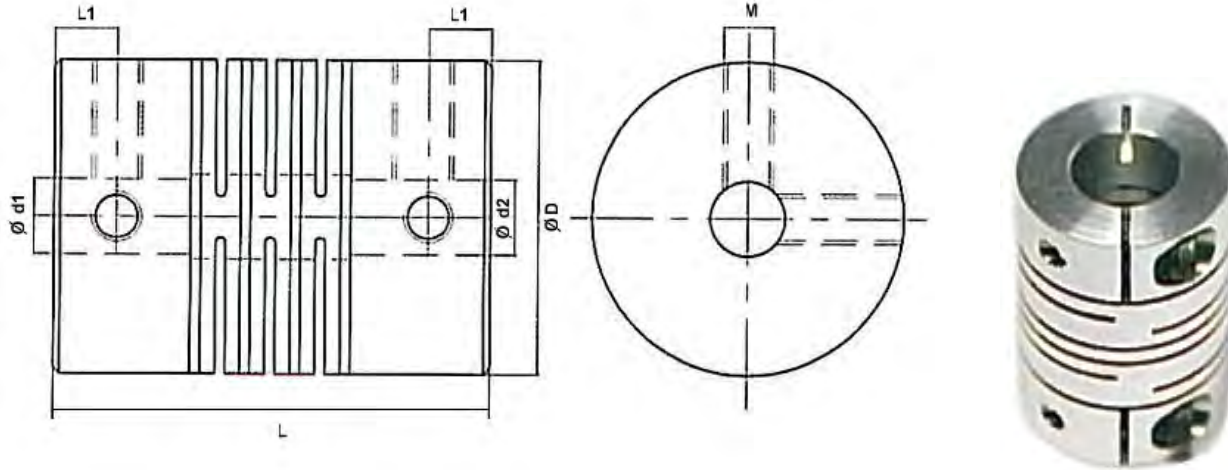
Specifications and sizes (mm) [note that 6.35mm = ¼"]

Part #	Length (L)	Diameter (D)	d1	d2	Rated Torque Max Torque	Eccentricity Error	Shaft Angle
DR1-C-20X25-5X5	25	20	5	5	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-5X6.35	30	25	5	6.35	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-6X6.35	30	25	6	6.35	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-8X5	30	25	8	5	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-8X6.35	30	25	8	6.35	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-10X5	30	25	10	5	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-10X6	30	25	10	6	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-10X6.35	30	25	10	6.35	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-10X8	30	25	10	8	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-12X5	30	25	12	5	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-12X6.35	30	25	12	6.35	1.5 N.m. 3.0 N.m.	±0.2mm	2°
DR1-C-25X30-12X8	30	25	12	8	1.5 N.m. 3.0 N.m.	±0.2mm	2°
All DR1-C couplings rated for 15,000 max. RPM.							

Note: We will be adding additional sizes in this style.

Special sizes can be supplied via special order (minimum quantities apply).

BR Flexible Motor Couplings (aluminum construction)

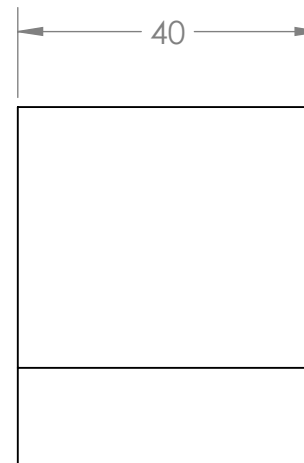
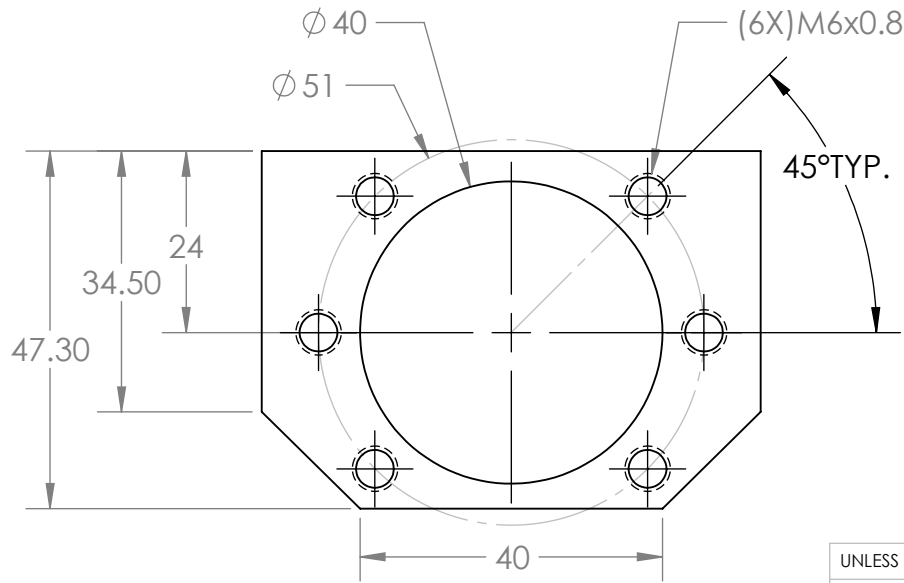
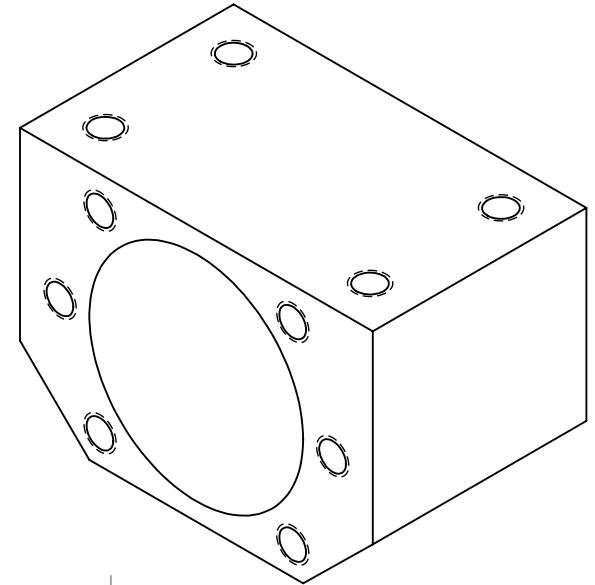
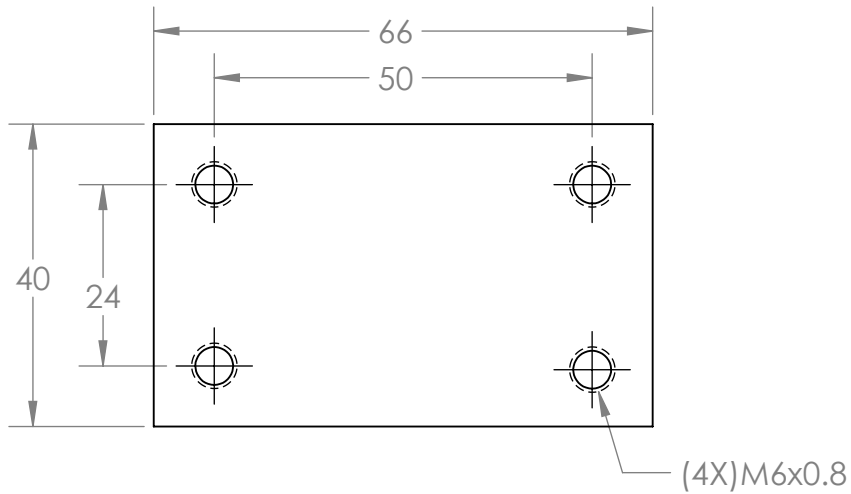


The BR series flexible motor coupling line provides an economical way to connect motors to ball screws and other shafts via clamping pressure applied by two set screws. Inexpensive, basic couplings suitable for many applications.

Specifications and sizes (mm) [note that 6.35mm = ¼"]

Part #	Length (L)	Diameter (D)	d1	d2	Rated Torque Max Torque	Eccentricity Error	Shaft Angle
BR-20X25-5X5	25	20	5	5	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-5X6.35	30	25	5	6.35	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-6X6.35	30	25	6	6.35	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-8X5	30	25	8	5	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-8X6.35	30	25	8	6.35	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-10X5	30	25	10	5	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-10X6	30	25	10	6	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-10X6.35	30	25	10	6.35	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-10X8	30	25	10	8	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-12X5	30	25	12	5	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-12X6.35	30	25	12	6.35	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR-25X30-12X8	30	25	12	8	0.5 N.m. 1.0 N.m.	±0.2mm	2°
BR couplings rated for 15,000 max. RPM.							

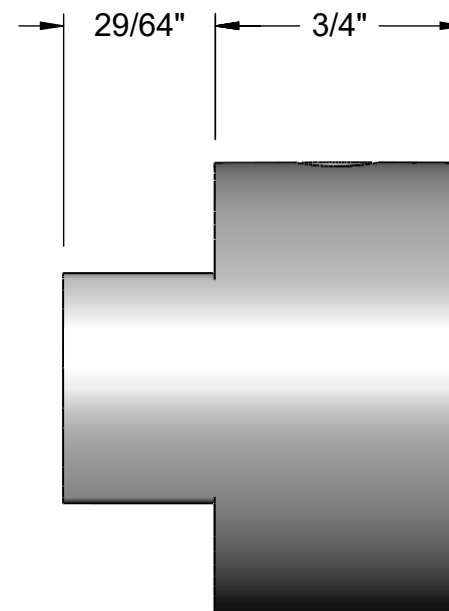
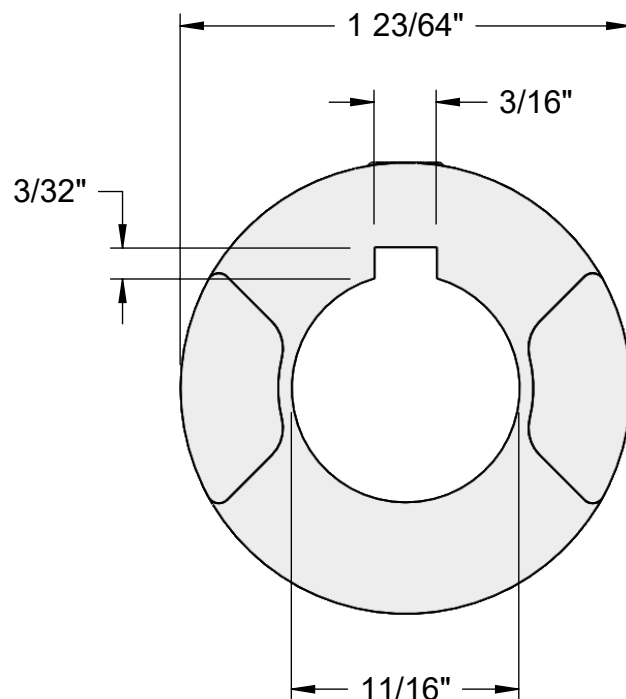
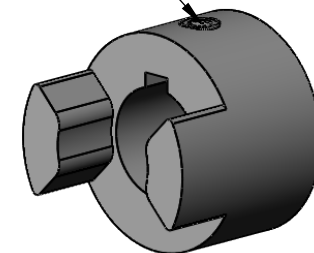
Special sizes can be supplied via special order (minimum quantities apply).



PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS
DRAWING IS THE SOLE PROPERTY OF
<INSERT COMPANY NAME HERE>. ANY
REPRODUCTION IN PART OR AS A WHOLE
WITHOUT THE WRITTEN PERMISSION OF
<INSERT COMPANY NAME HERE> IS
PROHIBITED.

UNLESS OTHERWISE SPECIFIED:		NAME	DATE	TITLE: <div>115000: HD25 NUT BRACKET</div>			
DIMENSIONS ARE IN MILLIMETERS TOLERANCES: +/- .25mm	DRAWN:						
	CHECKED:						
	COMMENTS:				DWG. NO. <div>HD25</div>		REV
MATERIAL							
FINISH							
DO NOT SCALE DRAWING					SCALE: 1:1	WEIGHT:	SHEET 1 OF 1

1/4"-20 x 1/4" Set Screw



Complete Coupling (Two Hubs and One Spider) Overall Length 1 63/64"

SOLIDWORKS Educational Product. For Instructional Use Only.

McMASTER-CARR CAD

<http://www.mcmaster.com>
© 2013 McMaster-Carr Supply Company

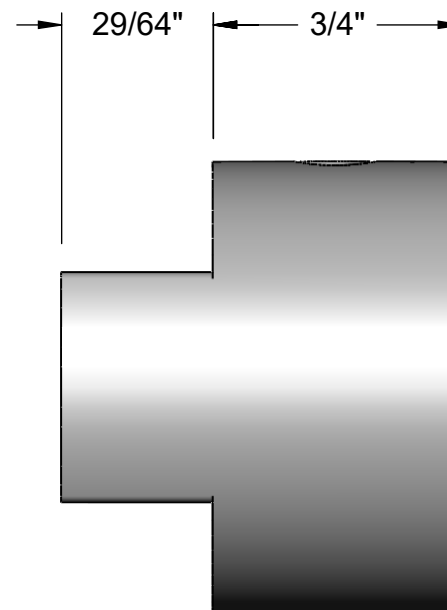
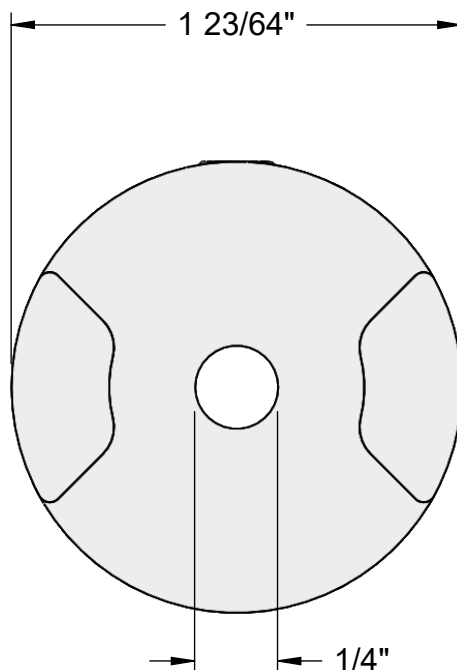
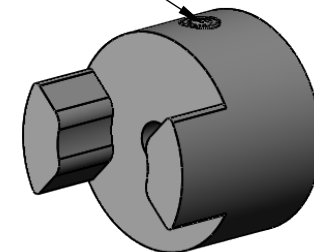
Information in this drawing is provided for reference only.

PART
NUMBER

116100

Coupling Hub for Replaceable-Center
Flexible Shaft Coupling

1/4"-20 x 5/16" Set Screw



Complete Coupling (Two Hubs and One Spider) Overall Length 1 63/64"

SOLIDWORKS Educational Product. For Instructional Use Only.

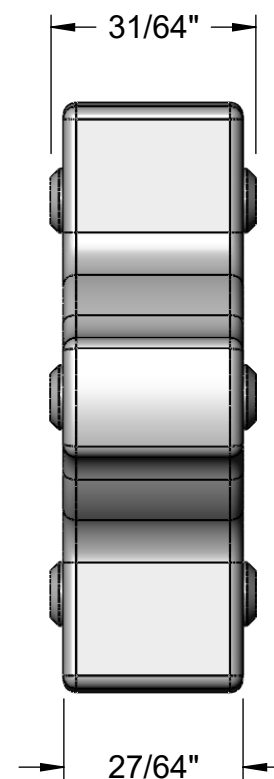
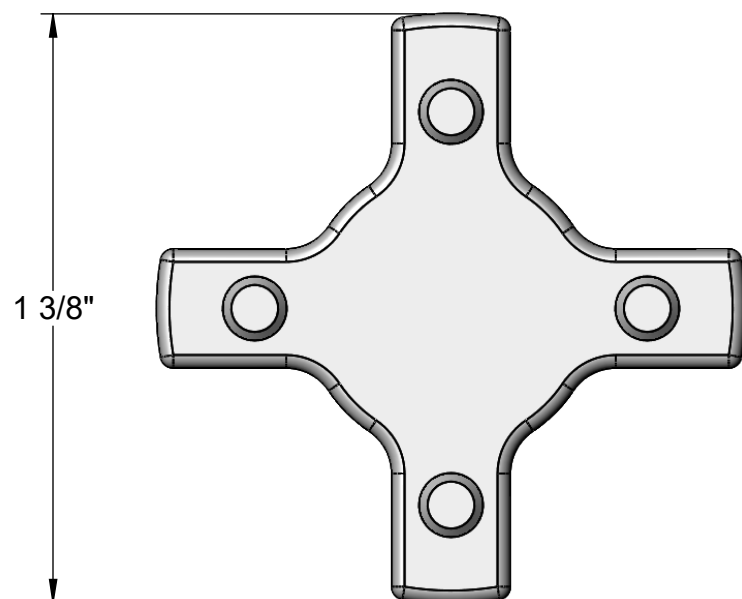
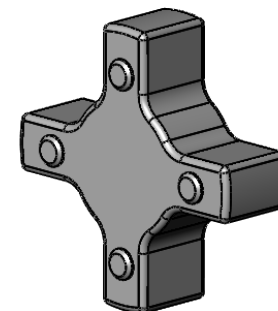
McMASTER-CARR CAD

<http://www.mcmaster.com>
© 2013 McMaster-Carr Supply Company
Information in this drawing is provided for reference only.

PART
NUMBER

116200

Coupling Hub for Replaceable-Center
Flexible Shaft Coupling



McMASTER-CARR CAD

<http://www.mcmaster.com>
© 2013 McMaster-Carr Supply Company

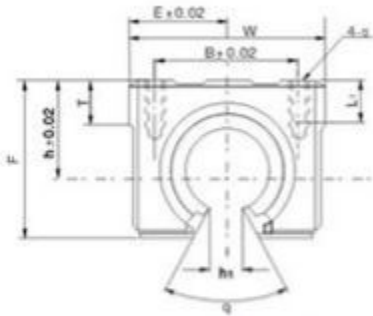
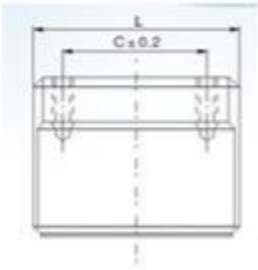
Information in this drawing is provided for reference only.

PART
NUMBER

116300

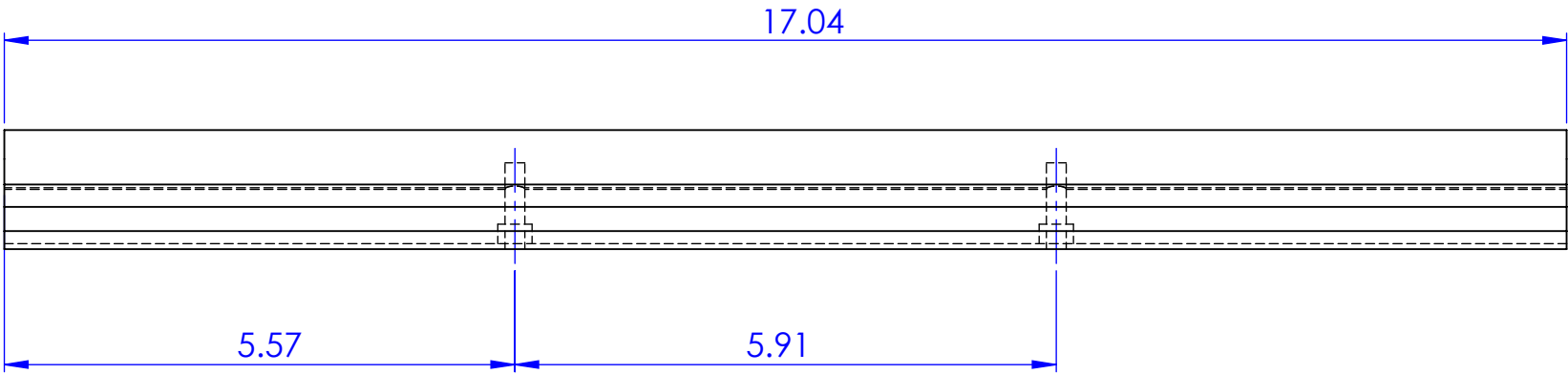
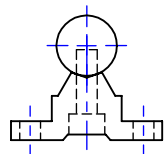
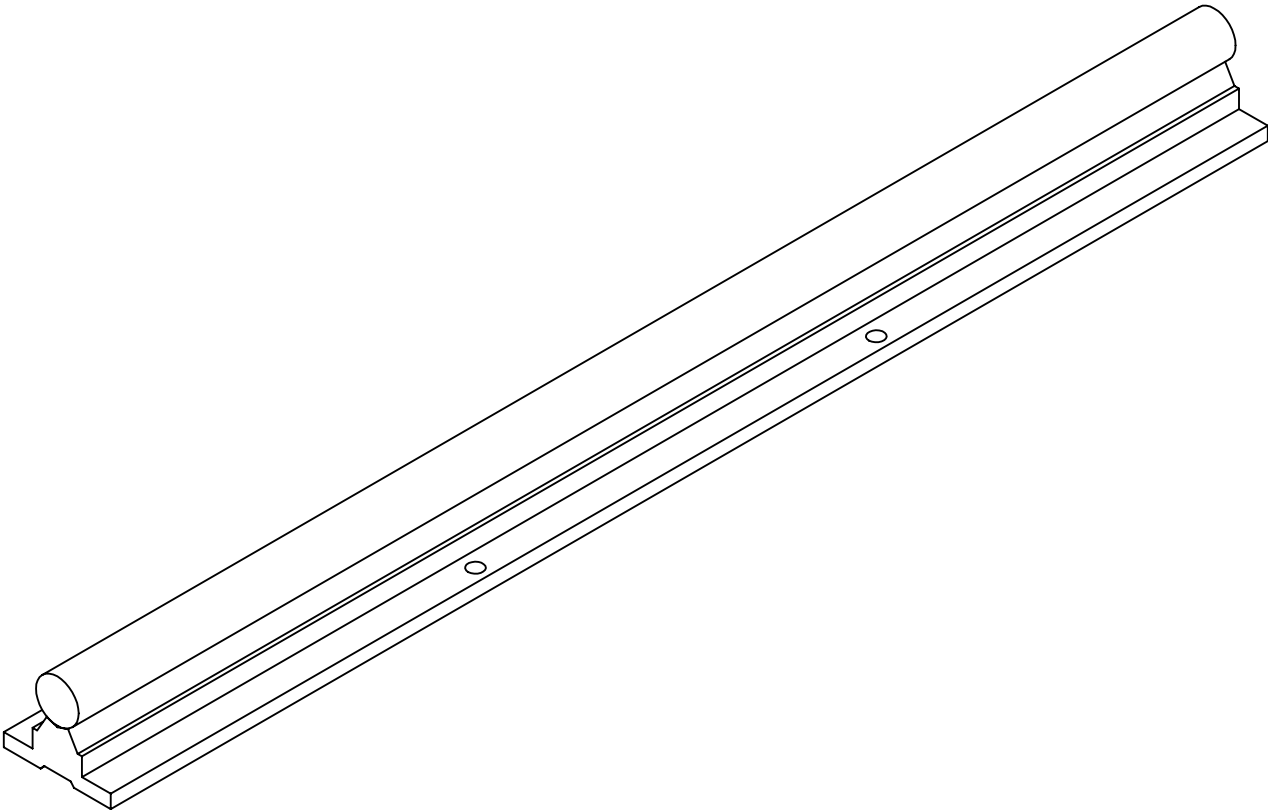
Spider for Replaceable-Center
Flexible Shaft Coupling

PART #: 121000

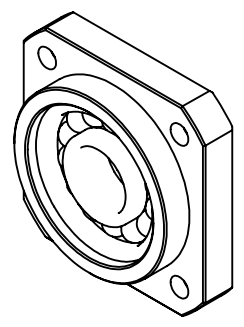
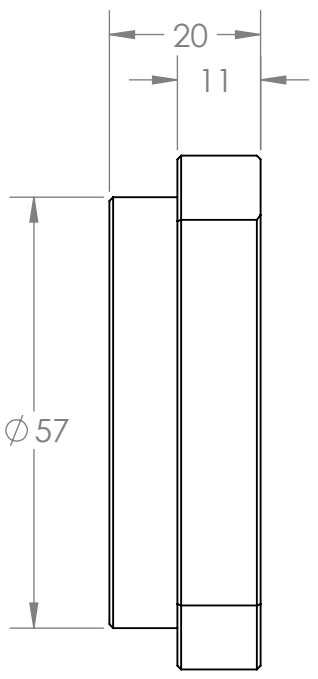
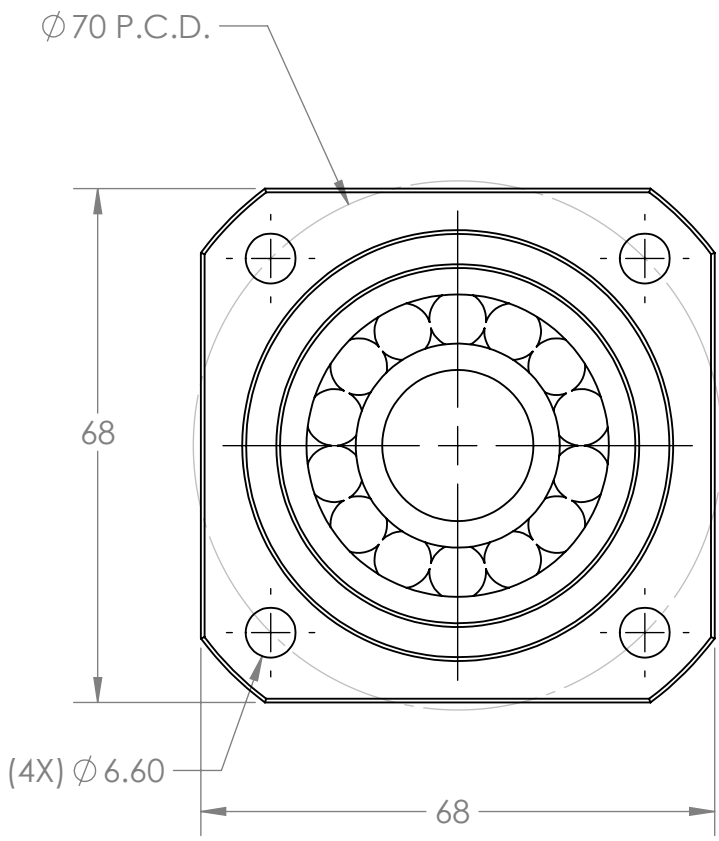


滑块型号 Unit Type	主要尺寸 Dimensions (mm)												配合直线轴承 Slide bush			重量 Weight (kg/m)
	h	E	W	L	F	h ₁	q	B	C	S	L ₁	T	型号 Type	基本负载率 Basic load rating		
														动 Dynamic C (kgf)	静 Static Co (kgf)	
SBR 10UU	15	18	36	32	24	6	80°	25	20	M5	10	7	LM10UU-OP	372	549	65
SBR 12UU	17.5	20.5	41	39	28	7.5	80°	28	26	M5	10	9	LM12UU-OP	420	610	100
SBR 13UU	17	20	40	39	27.6	8.5	80°	28	28	M5	10	8	LM13UU-OP	510	784	100
SBR 16UU	20	22.5	45	45	33	10	80°	32	30	M5	12	9	LM16UU-OP	774	1180	150
SBR 20UU	23	24	48	50	39	10	60°	35	35	M6	12	11	LM20UU-OP	882	1370	200
SBR 25UU	27	30	60	65	47	11.5	50°	40	40	M6	12	14	LM25UU-OP	980	1570	450
SBR 30UU	33	35	70	70	56	14	50°	50	50	M8	18	15	LM30UU-OP	1570	2740	630
SBR 35UU	37	40	80	80	63	16	50°	55	55	M8	18	18	LM35UU-OP	1670	3140	925
SBR 40UU	42	45	90	90	72	19	50°	65	65	M10	20	20	LM40UU-OP	2160	4020	1330
SBR 50UU	53	60	120	110	92	23	50°	80	80	M10	20	25	LM50UU-OP	3620	7940	3000

NOTES:
RAILS PURCHASED AND TO BE CUT TO LENGTH
UNLESS OTHERWISE SPECIFIED:
1. ALL DIMENSIONS IN INCHES
2. TOLERANCES
 X.XX = $\pm .01$
 ANGLES = $\pm 2^\circ$
3. RAIL MASS = 4.24 lb

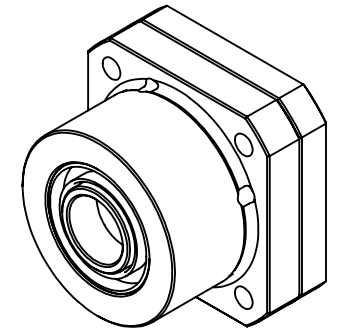
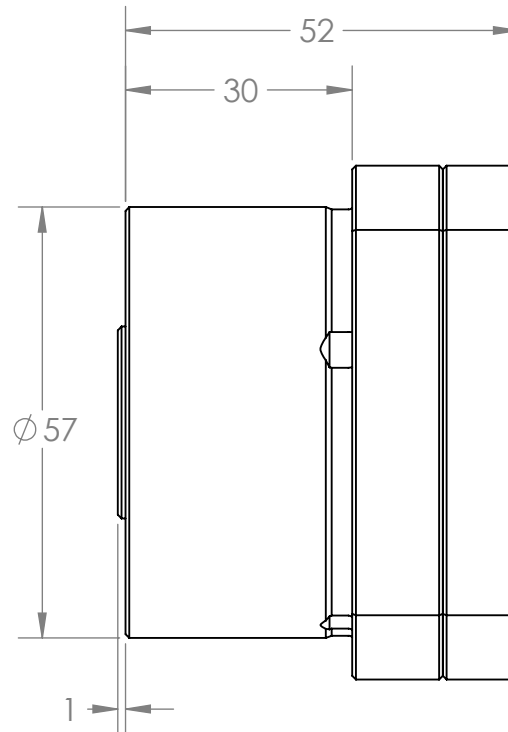
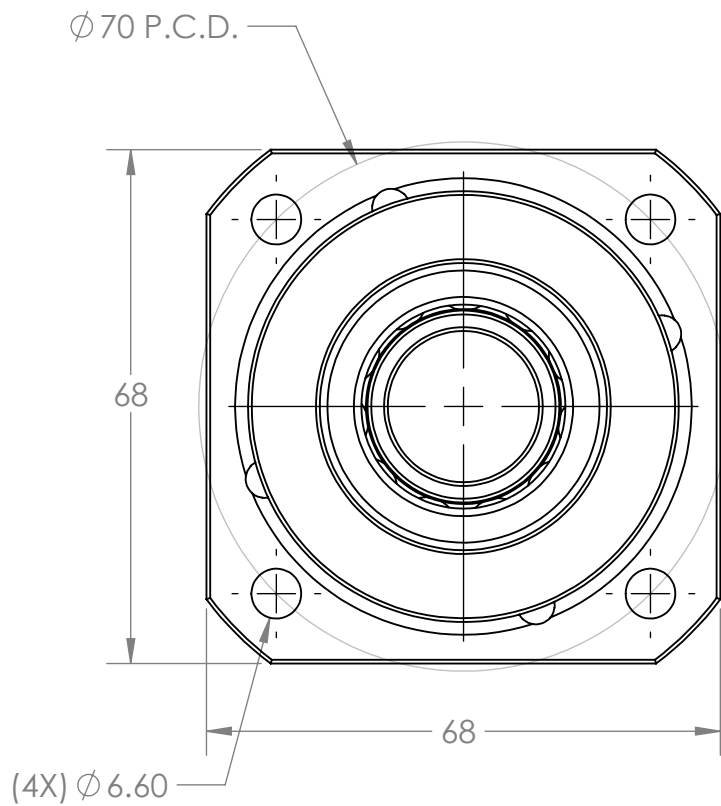


Cal Poly Mechanical Engineering ME 429 - Winter 2021	CNC FEED DRIVE	CDR MODEL	Title: SBR16 LINEAR RAILS		Drwn. By: Nick De Simone
	Part #: 133000	Nxt Asb: ASSEMBLY	Date: 2/11/21	Scale: 1=2	Chkd. By: ME STAFF



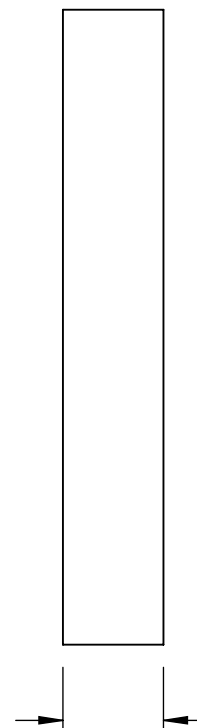
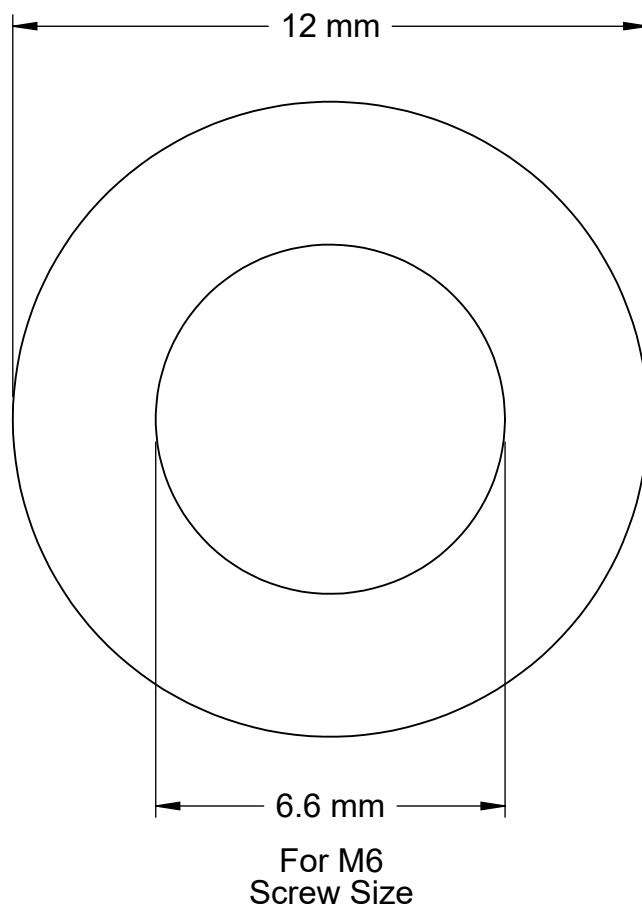
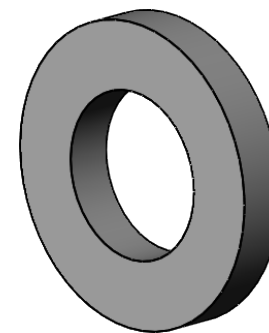
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS
DRAWING IS THE SOLE PROPERTY OF
<INSERT COMPANY NAME HERE>. ANY
REPRODUCTION IN PART OR AS A WHOLE
WITHOUT THE WRITTEN PERMISSION OF
<INSERT COMPANY NAME HERE> IS
PROHIBITED.

UNLESS OTHERWISE SPECIFIED:		NAME	DATE	TITLE: 122000: WBF-20 END SUPPORT	
	DRAWN:				
	CHECKED:				
				DWG. NO. WBF-20	
MATERIAL	COMMENTS:			REV	
FINISH					
DO NOT SCALE DRAWING					
SCALE: 1:2		WEIGHT:		SHEET 1 OF 1	



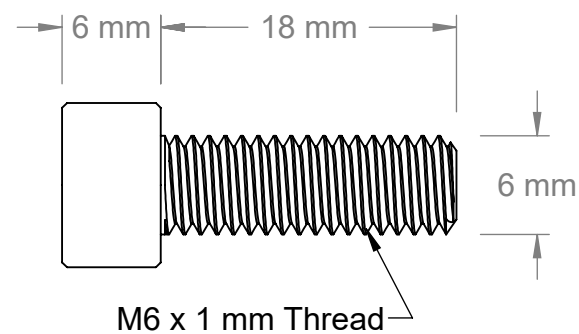
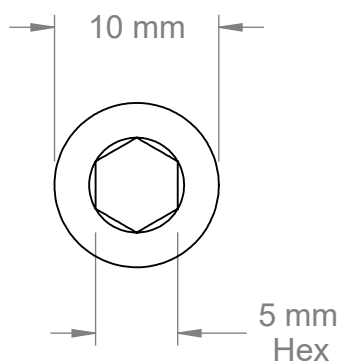
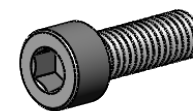
PROPRIETARY AND CONFIDENTIAL
 THE INFORMATION CONTAINED IN THIS
 DRAWING IS THE SOLE PROPERTY OF
 <INSERT COMPANY NAME HERE>. ANY
 REPRODUCTION IN PART OR AS A WHOLE
 WITHOUT THE WRITTEN PERMISSION OF
 <INSERT COMPANY NAME HERE> IS
 PROHIBITED.

UNLESS OTHERWISE SPECIFIED:		NAME	DATE
	DRAWN:		
	CHECKED:		
	COMMENTS:		
MATERIAL			
FINISH			
DO NOT SCALE DRAWING			
TITLE:		123000: WBK-20 END SUPPORT	
DWG. NO.		REV	
WBK-20			
SCALE: 1:2	WEIGHT:	SHEET 1 OF 1	



Washer may vary from
1.3 mm to 1.9 mm in thickness.

McMASTER-CARR <small>CAD</small>	PART NUMBER	124000
http://www.mcmaster.com © 2020 McMaster-Carr Supply Company Information in this drawing is provided for reference only.	Metric General Purpose Washer	



McMASTER-CARR CAD

<http://www.mcmaster.com>
© 2014 McMaster-Carr Supply Company

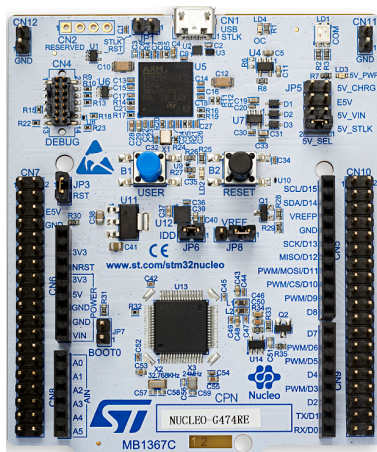
Information in this drawing is provided for reference only.

PART
NUMBER

131000

Metric Alloy Steel
Socket Head Cap Screw

STM32 Nucleo-64 boards



NUCLEO-G474RE example. Boards with different references show different layouts. Picture is not contractual.

Features

- Common features
 - STM32 microcontroller in LQFP64 package
 - 1 user LED shared with ARDUINO®
 - 1 user and 1 reset push-buttons
 - 32.768 kHz crystal oscillator
 - Board connectors:
 - ARDUINO® Uno V3 expansion connector
 - ST morpho extension pin headers for full access to all STM32 I/Os
 - Flexible power-supply options: ST-LINK, USB V_{BUS} , or external sources
 - On-board ST-LINK debugger/programmer with USB re-enumeration capability: mass storage, Virtual COM port and debug port
 - Comprehensive free software libraries and examples available with the STM32Cube MCU Package
 - Support of a wide choice of Integrated Development Environments (IDEs) including IAR Embedded Workbench®, MDK-ARM, and STM32CubeIDE
- Board-specific features
 - External SMPS to generate V_{core} logic supply
 - 24 MHz HSE
 - Board connectors:
 - External SMPS experimentation dedicated connector
 - Micro-AB or Mini-AB USB connector for the ST-LINK
 - MIP1® debug connector
 - Arm® Mbed Enabled™ compliant

Description

The STM32 Nucleo-64 board provides an affordable and flexible way for users to try out new concepts and build prototypes by choosing from the various combinations of performance and power consumption features, provided by the STM32 microcontroller. For the compatible boards, the external SMPS significantly reduces power consumption in Run mode.

The ARDUINO® Uno V3 connectivity support and the ST morpho headers allow the easy expansion of the functionality of the STM32 Nucleo open development platform with a wide choice of specialized shields.

The STM32 Nucleo-64 board does not require any separate probe as it integrates the ST-LINK debugger/programmer.

The STM32 Nucleo-64 board comes with the STM32 comprehensive free software libraries and examples available with the STM32Cube MCU Package.

Product status link

NUCLEO-XXXXRX

NUCLEO-F030R8, NUCLEO-F070RB, NUCLEO-F072RB, NUCLEO-F091RC, NUCLEO-F103RB, NUCLEO-F302R8, NUCLEO-F303RE, NUCLEO-F334R8, NUCLEO-F401RE, NUCLEO-F410RB, NUCLEO-F411RE, NUCLEO-F446RE, NUCLEO-G070RB, NUCLEO-G071RB, NUCLEO-G0B1RE, NUCLEO-G431RB, NUCLEO-G474RE, NUCLEO-G491RE, NUCLEO-L010RB, NUCLEO-L053R8, NUCLEO-L073RZ, NUCLEO-L152RE, NUCLEO-L452RE, NUCLEO-L476RG.

NUCLEO-XXXXRX-P

NUCLEO-L412RB-P, NUCLEO-L433RC-P, NUCLEO-L452RE-P.






1 Ordering information

To order an STM32 Nucleo-64 board, refer to [Table 1](#). For a detailed description of each board, refer to its user manual on the product web page. Additional information is available from the datasheet and reference manual of the target STM32.

Table 1. List of available products

Order code	Board reference	User manual	Target STM32	Differentiating features
NUCLEO-F030R8	MB1136	UM1724	STM32F030R8T6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F070RB			STM32F070RBT6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F072RB			STM32F072RBT6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F091RC			STM32F091RCT6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F103RB			STM32F103RBT6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F302R8			STM32F302R8T6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F303RE			STM32F303RET6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F334R8			STM32F334R8T6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F401RE			STM32F401RET6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F410RB			STM32F410RBT6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F411RE			STM32F411RET6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-F446RE			STM32F446RET6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-G070RB	MB1360	UM2324	STM32G070RBT6	<ul style="list-style-type: none"> ST-LINK/V2-1 on Micro-AB USB connector
NUCLEO-G071RB			STM32G071RBT6	<ul style="list-style-type: none"> ST-LINK/V2-1 on Micro-AB USB connector



NUCLEO-XXXXRX NUCLEO-XXXXRX-P

Ordering information

Order code	Board reference	User manual	Target STM32	Differentiating features
NUCLEO-G0B1RE	MB1360	UM2324	STM32G0B1RET6	<ul style="list-style-type: none"> ST-LINK/V2-1 on Micro-AB USB connector
NUCLEO-G431RB	MB1367	UM2505	STM32G431RBT6U	<ul style="list-style-type: none"> STLINK-V3E on Micro-AB USB connector 24 MHz HSE MIPI® debug connector
NUCLEO-G474RE			STM32G474RET6U	<ul style="list-style-type: none"> STLINK-V3E on Micro-AB USB connector 24 MHz HSE MIPI® debug connector
NUCLEO-G491RE			STM32G491RET6U	<ul style="list-style-type: none"> STLINK-V3E on Micro-AB USB connector 24 MHz HSE MIPI® debug connector
NUCLEO-L010RB	MB1136	UM1724	STM32L010RBT6	<ul style="list-style-type: none"> ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-L053R8			STM32L053R8T6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-L073RZ			STM32L073RZT6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-L152RE			STM32L152RET6	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-L412RB-P	MB1319	UM2206	STM32L412RBT6PU	<ul style="list-style-type: none"> ST-LINK/V2-1 on Micro-AB USB connector External SMPS
NUCLEO-L433RC-P			STM32L433RCT6PU	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Micro-AB USB connector External SMPS
NUCLEO-L452RE	MB1136	UM1724	STM32L452RET6U	<ul style="list-style-type: none"> ST-LINK/V2-1 on Mini-AB USB connector
NUCLEO-L452RE-P	MB1319	UM2206	STM32L452RET6PU	<ul style="list-style-type: none"> ST-LINK/V2-1 on Micro-AB USB connector External SMPS
NUCLEO-L476RG	MB1136	UM1724	STM32L476RGT6U	<ul style="list-style-type: none"> Arm® Mbed Enabled™ ST-LINK/V2-1 on Mini-AB USB connector



1.1 Product marking

The sticker located on the top or bottom side of the PCB board shows the information about product identification such as board reference, revision, and serial number.

The first identification line has the following format: "MBxxxx-Variant-yyz", where "MBxxxx" is the board reference, "Variant" (optional) identifies the mounting variant when several exist, "y" is the PCB revision and "zz" is the assembly revision: for example B01.

The second identification line is the board serial number used for traceability.

Evaluation tools marked as "ES" or "E" are not yet qualified and therefore not ready to be used as reference design or in production. Any consequences deriving from such usage will not be at ST charge. In no event, ST will be liable for any customer usage of these engineering sample tools as reference designs or in production.

"E" or "ES" marking examples of location:

- On the targeted STM32 that is soldered on the board (For an illustration of STM32 marking, refer to the STM32 datasheet "Package information" paragraph at the www.st.com website).
- Next to the evaluation tool ordering part number that is stuck or silk-screen printed on the board.

Some boards feature a specific STM32 device version, which allows the operation of any bundled commercial stack/library available. This STM32 device shows a "U" marking option at the end of the standard part number and is not available for sales.

In order to use the same commercial stack in his application, a developer may need to purchase a part number specific to this stack/library. The price of those part numbers includes the stack/library royalties.

1.2 Codification

The meaning of the codification is explained in Table 2.

Table 2. Codification explanation

NUCLEO-XXYYRT NUCLEO-XXYYRT-P	Description	Example: NUCLEO-L452RE
XX	MCU series in STM32 Arm Cortex MCUs	STM32L4 Series
YY	MCU product line in the series	STM32L452
R	STM32 package pin count	64 pins
T	STM32 Flash memory size: <ul style="list-style-type: none"> • 8 for 64 Kbytes • B for 128 Kbytes • C for 256 Kbytes • E for 512 Kbytes • G for 1 Mbyte • Z for 192 Kbytes 	512 Kbytes
-P	STM32 has external SMPS function	No SMPS



2 Development environment

2.1 System requirements

- Windows® OS (7, 8 and 10), Linux® 64-bit, or macOS®
- USB Type-A or USB Type-C® to Micro-B cable, or USB Type-A or USB Type-C® to Mini-B cable (depending on the board reference)

Note: macOS® is a trademark of Apple Inc. registered in the U.S. and other countries.
All other trademarks are the property of their respective owners.

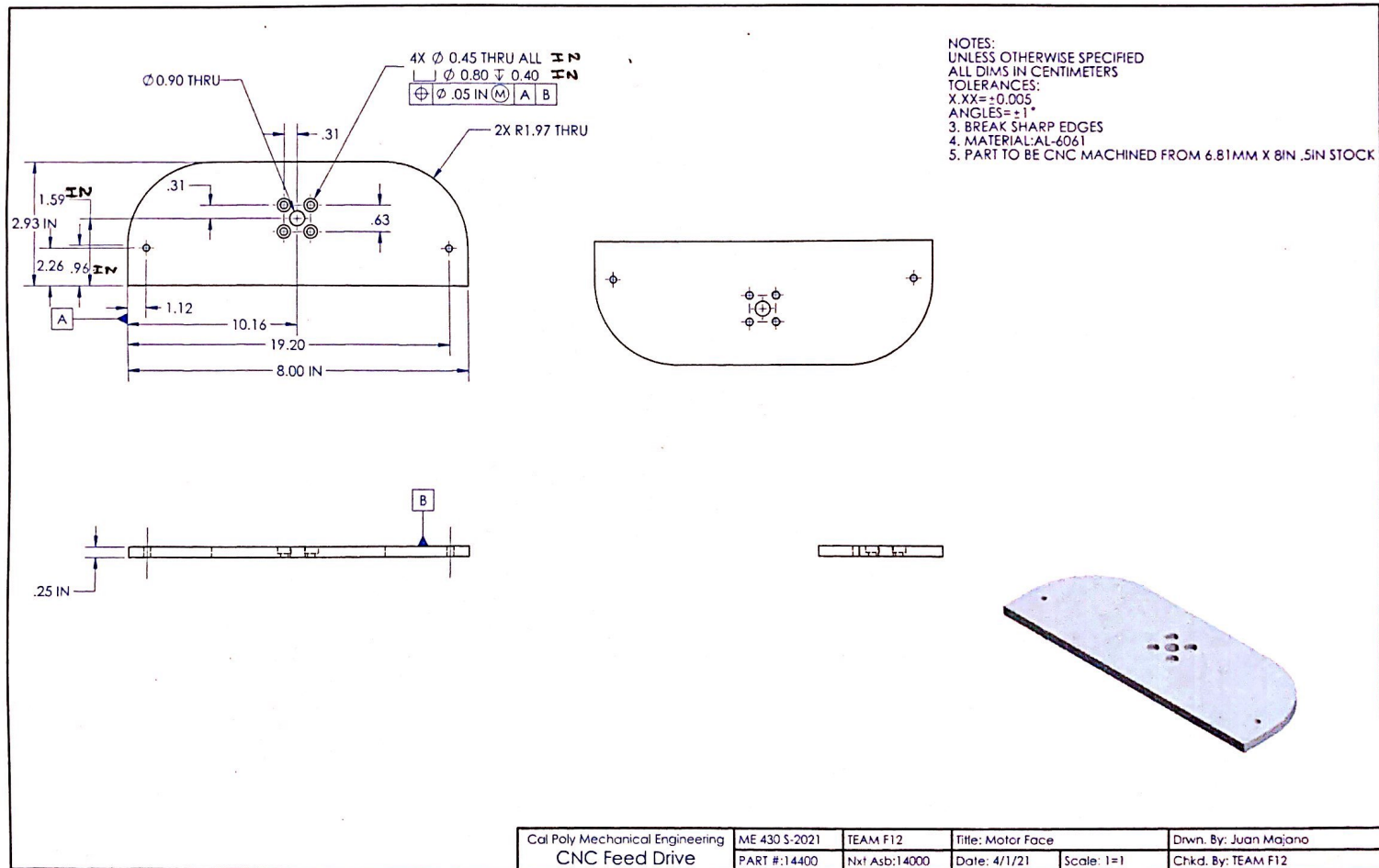
2.2 Development toolchains

- IAR Systems - IAR Embedded Workbench®⁽¹⁾
- Keil® - MDK-ARM⁽¹⁾
- STMicroelectronics - STM32CubeIDE
- Arm® - Mbed Studio^{(2) (3)}

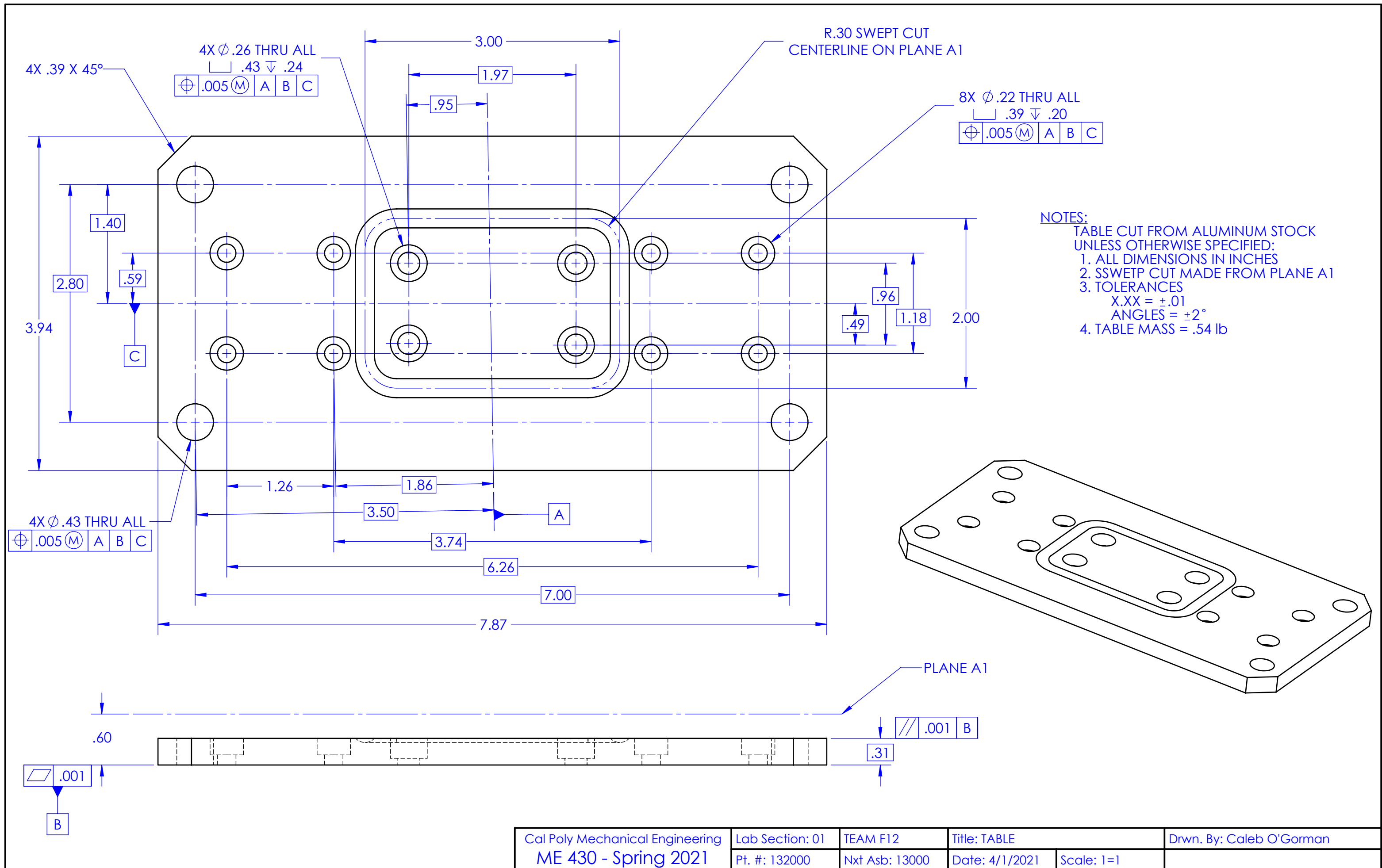
1. On Windows® only.
2. Arm and Mbed are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and or elsewhere.
3. Refer to the os.mbed.com website and to the "Ordering information" section to determine which order codes are supported.

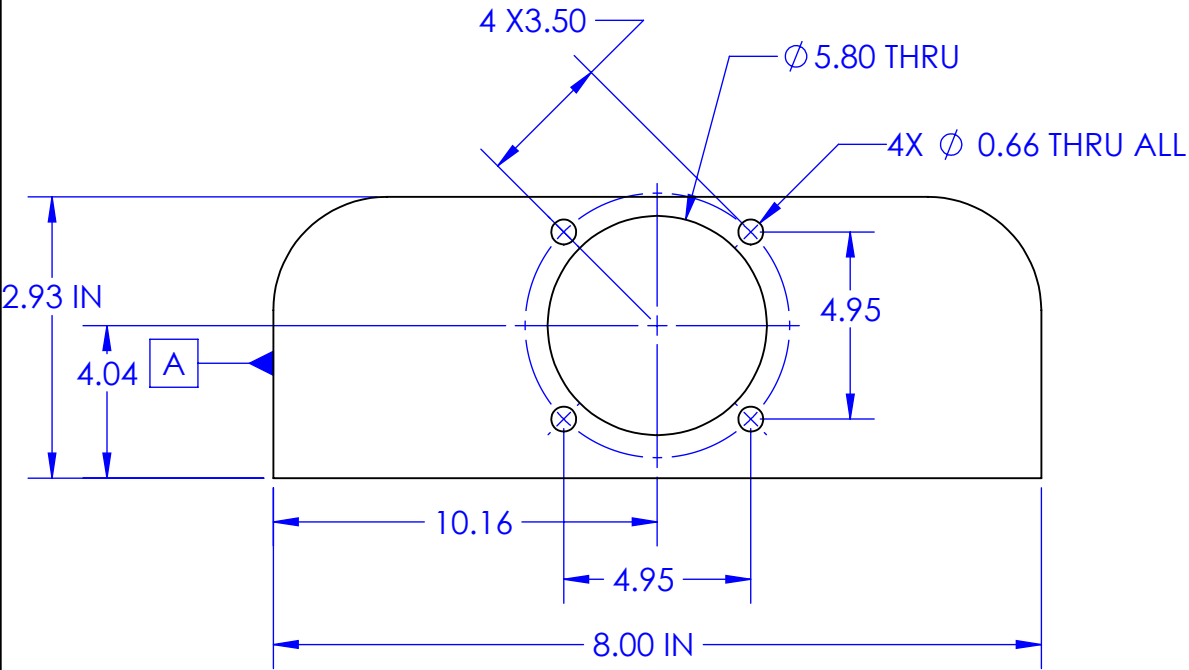
2.3 Demonstration software

The demonstration software, included in the STM32Cube MCU Package corresponding to the on-board microcontroller, is preloaded in the STM32 Flash memory for easy demonstration of the device peripherals in standalone mode. The latest versions of the demonstration source code and associated documentation can be downloaded from www.st.com.

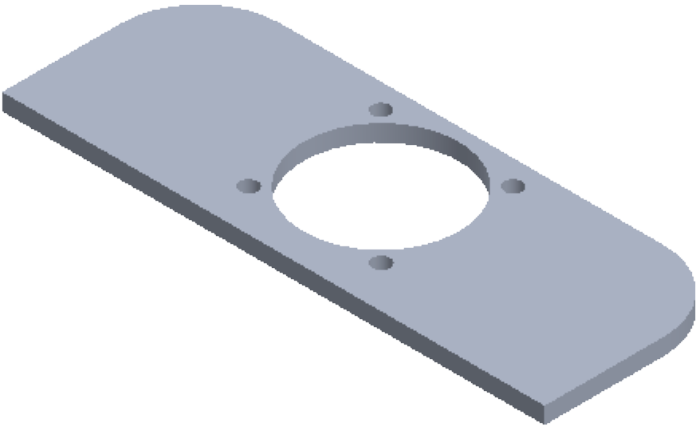
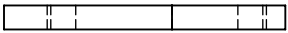
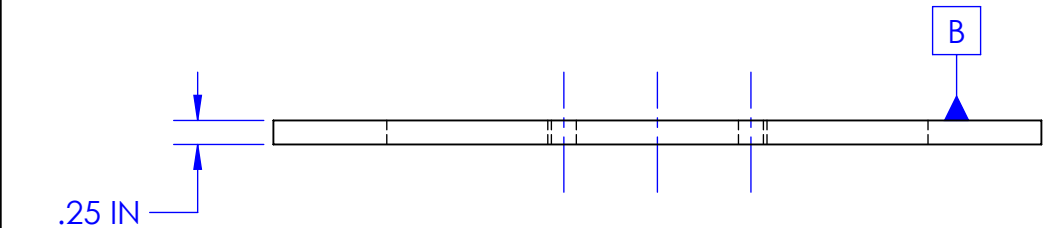


SOLIDWORKS Educational Product. For Instructional Use Only.

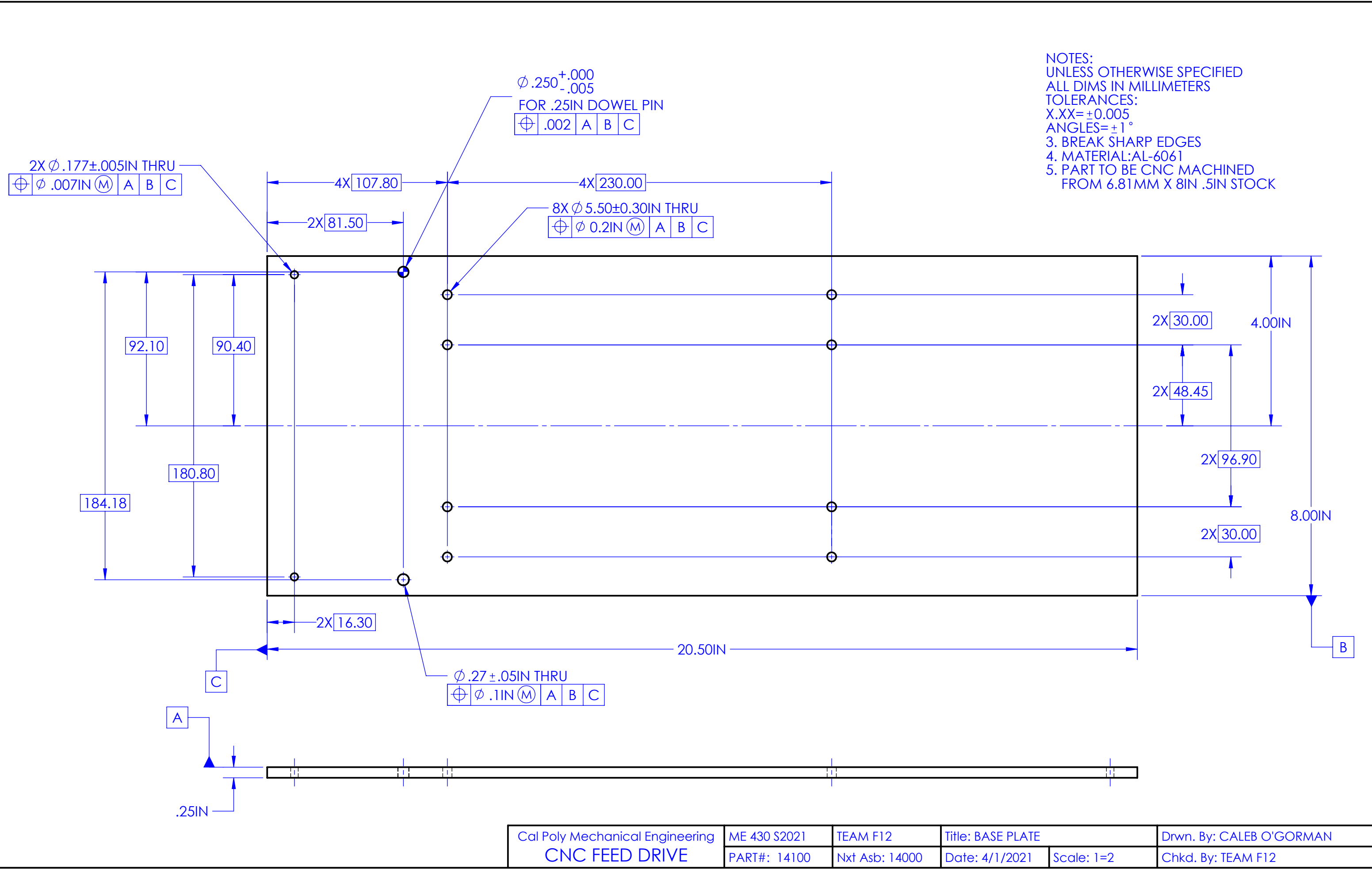


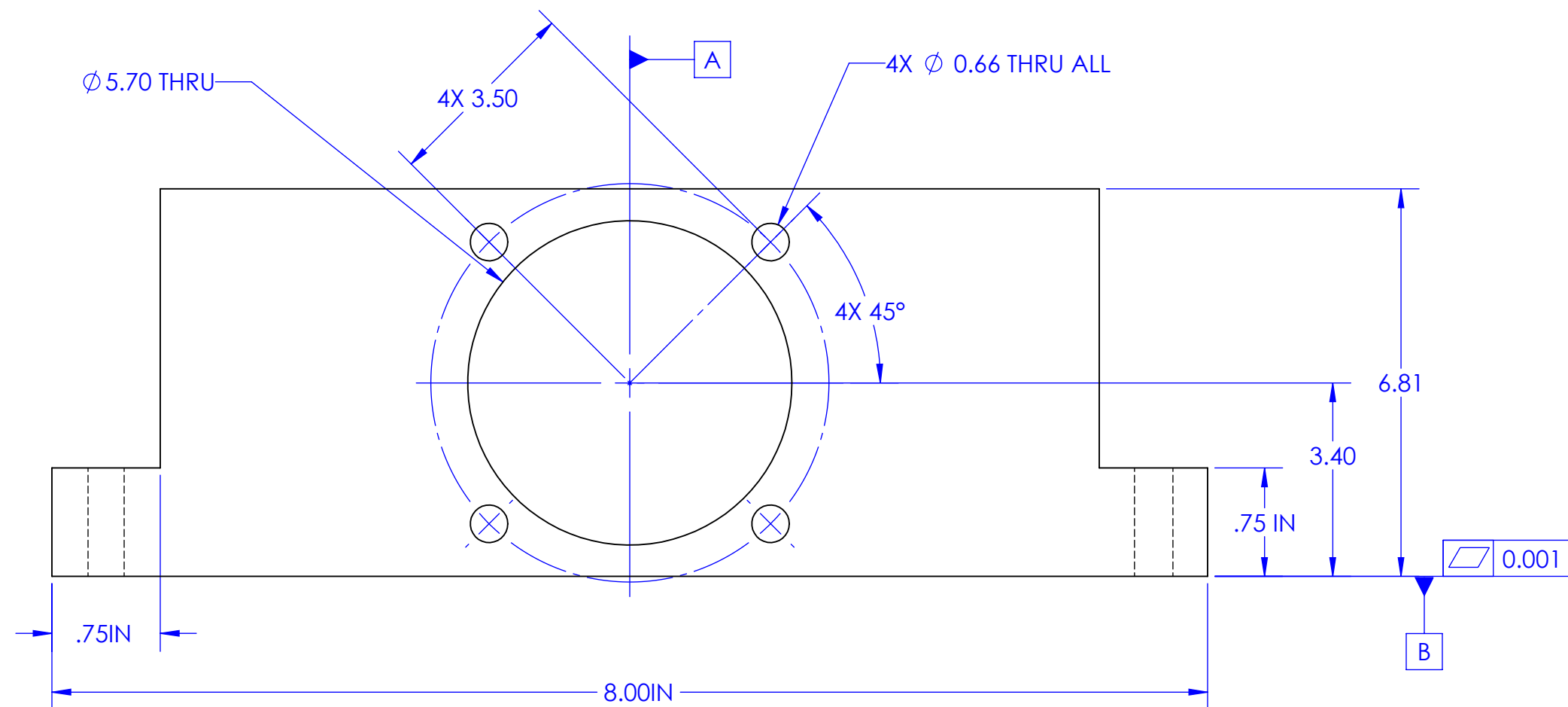
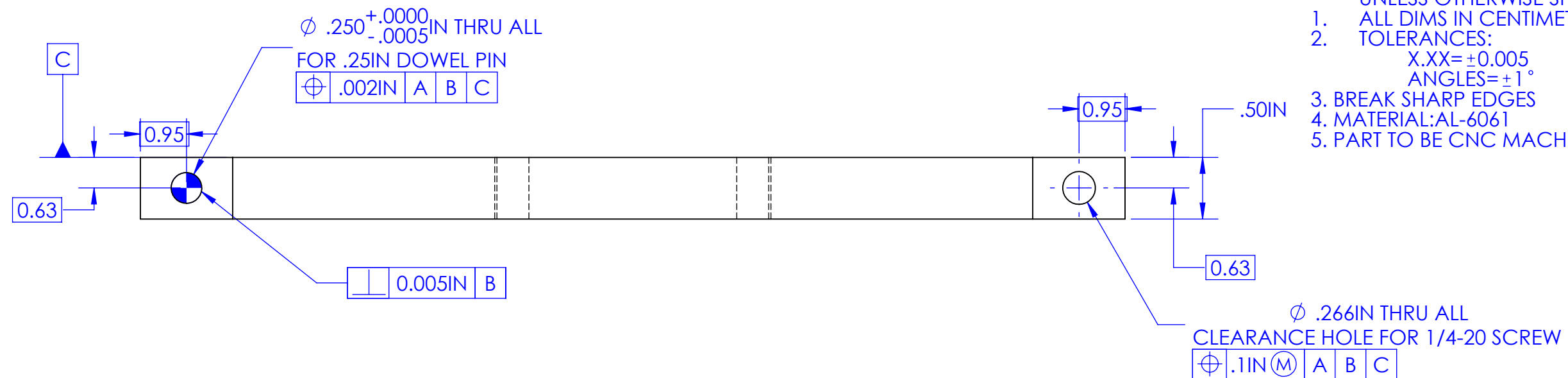


NOTES:
UNLESS OTHERWISE SPECIFIED
ALL DIMS IN CENTIMETERS
TOLERANCES:
X.XX=±0.005
ANGLES=±1°
3. BREAK SHARP EDGES
4. MATERIAL:AL-6061
5. PART TO BE CNC MACHINED FROM 6.81MM X 8IN .5IN STOCK



Cal Poly Mechanical Engineering CNC Feed Drive	ME 430 S-2021	Team F12	Title: Ball Bearing Face Mount		Drwn. By: Juan Majano
	PART #:14500	NXt Asb: 14000	Date: 4/1/21	Scale: 1=1	Chkd. By: TEAM F12





Cal Poly Mechanical Engineering
CNC FEED DRIVE

ME 430 S2021
PART#: 14300

TEAM F12
Nxt Asb: 14000

Title: DATUM FACE
Date: 4/1/21

Scale: 1=1

Drwn. By: RYAN FUNCHES
Chkd. by: TEAM F12

A.16 Risk Assessment

F12_CNC Feed Drive

2/18/2021

designsafe Report

Application: F12_CNC Feed Drive Analyst Name(s): Caleb O'Gorman, Juan Majano, Ryan Funchess, Nick DeSimone, and Samuel Wong
 Description: Company: Cal Poly Mechanical Engineering
 Product Identifier: Facility Location:
 Assessment Type: Detailed
 Limits:
 Sources:
 Risk Scoring System: ANSI B11.0 (TR3) Two Factor
 Guide sentence: When doing [task], the [user] could be injured by the [hazard] due to the [failure mode].

Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods /Control System	Final Assessment		Status / Responsible /Comments /Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
1	All Users <None>	<None>						
2-1-1	Team Common Tasks	mechanical : pinch point User putting body part to close to moving table.	Serious Likely	High	Putting clear cover over moving parts	Serious Unlikely	Medium	TBD [3/19/2021] juan
2-1-2	Team Common Tasks	mechanical : product instability table placement	Serious Unlikely	Medium	Secure table	Serious Unlikely	Medium	TBD [2/19/2021] nick
2-1-3	Team Common Tasks	electrical / electronic : shorts / arcing / sparking broken wires	Moderate Unlikely	Low	check electrical weekly	Moderate Unlikely	Low	TBD [2/19/2021] ryan
2-1-4	Team Common Tasks	electrical / electronic : improper wiring assembly oversight	Serious Unlikely	Medium	Double-check during assembly	Serious Unlikely	Medium	TBD [2/19/2021] juan
2-1-5	Team Common Tasks	electrical / electronic : unexpected start up / motion controller issue	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD [2/19/2021] ryan
2-1-6	Team Common Tasks	electrical / electronic : overvoltage /overcurrent controller issue	Serious Likely	High	E-stop control	Serious Unlikely	Medium	TBD [2/19/2021] caleb

Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods /Control System	Final Assessment		Status / Responsible /Comments /Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
2-1-7	Team Common Tasks	electrical / electronic : power supply interruption improper wiring	Serious Unlikely	Medium	double-check during assembly	Serious Unlikely	Medium	TBD [2/19/2021] nick
2-1-8	Team Common Tasks	ergonomics / human factors : lifting / bending / twisting moving system	Minor Unlikely	Negligible	cart	Minor Unlikely	Negligible	TBD [3/19/2021] ryan
2-1-9	Team Common Tasks	noise / vibration : noise / sound levels > 80 dBA overrunning motor	Moderate Unlikely	Low	E-stop control	Moderate Unlikely	Low	TBD [3/19/2021] juan
2-1-10	Team Common Tasks	noise / vibration : noise / sound levels >120 dBA instantaneous overrunning motor	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD [3/19/2021] caleb
2-1-11	Team Common Tasks	noise / vibration : product / equipment damage falling off table	Serious Unlikely	Medium	adjustable enclosures / barriers	Serious Unlikely	Medium	TBD [3/19/2021] nick
2-1-12	Team Common Tasks	noise / vibration : interference with communications controller issue	Moderate Unlikely	Low	E-stop control	Moderate Unlikely	Low	TBD [3/19/2021] caleb
2-2-1	Team first use / test	mechanical : drawing-in / trapping / entanglement hair too close to system	Serious Unlikely	Medium	fixed enclosures / barriers	Serious Unlikely	Medium	TBD [3/19/2021] ryan
2-2-2	Team first use / test	mechanical : unexpected start controller/wiring issue	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD [3/19/2021] juan
2-2-3	Team first use / test	mechanical : Projectile table could launch masses	Moderate Unlikely	Low	fixed enclosures / barriers	Moderate Unlikely	Low	TBD [3/19/2021] nick

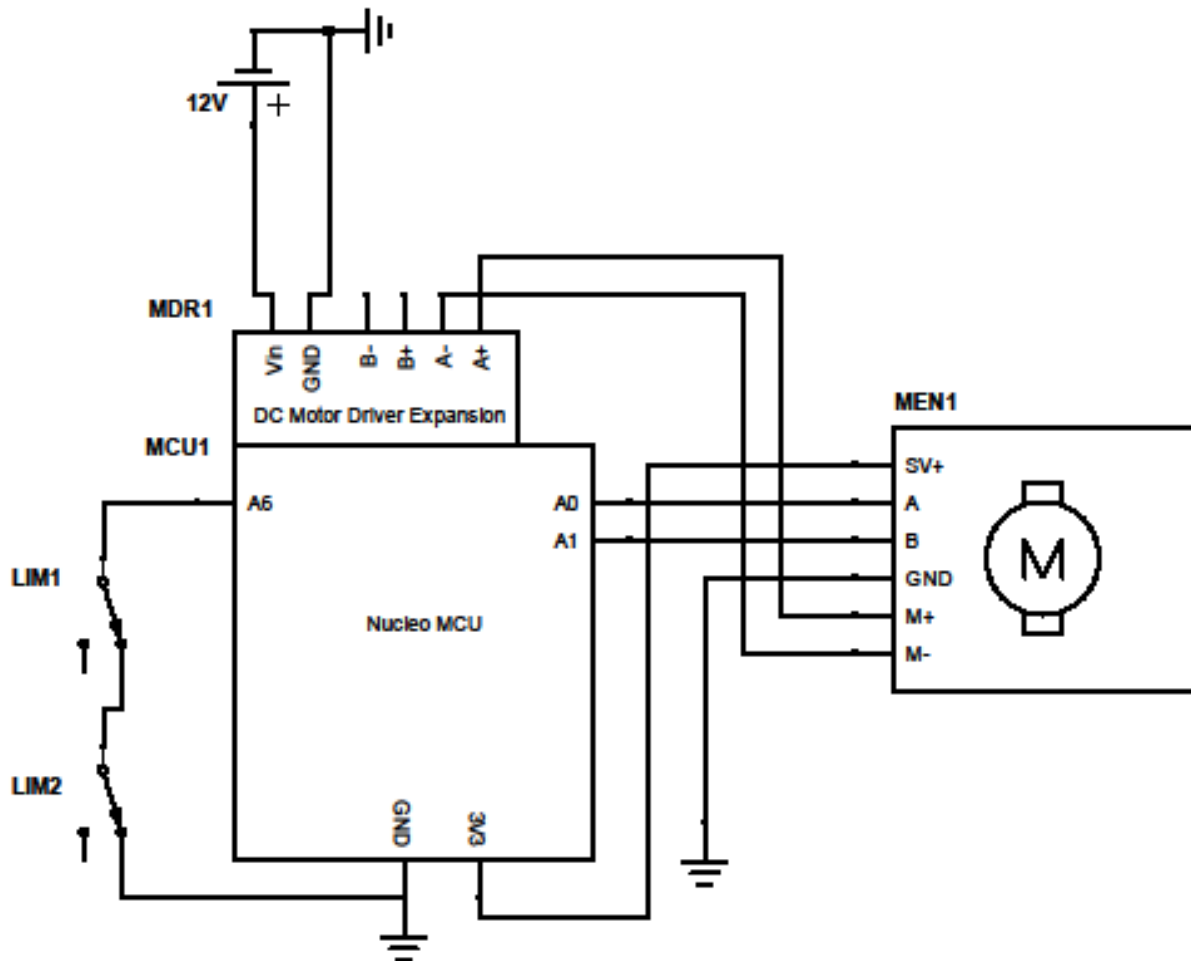
Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods /Control System	Final Assessment		Status / Responsible /Comments /Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
2-2-4	Team first use / test	electrical / electronic : shorts / arcing / sparking wiring issue	Serious Unlikely	Medium	E-stop control	Serious Remote	Low	TBD [3/19/2021] caleb
2-2-5	Team first use / test	electrical / electronic : improper wiring incorrect assembly	Serious Unlikely	Medium	Double-check assembly	Serious Unlikely	Medium	TBD [3/19/2021] ryan
2-2-6	Team first use / test	noise / vibration : noise / sound levels > 80 dBA motor malfunction	Moderate Unlikely	Low	E-stop control	Moderate Unlikely	Low	TBD [3/19/2021] caleb
2-2-7	Team first use / test	noise / vibration : noise / sound levels >120 dBA instantaneous motor malfunction	Moderate Unlikely	Low	E-stop control	Moderate Unlikely	Low	TBD [3/19/2021] nick
2-3	Team trouble-shooting / problem solving	<None>						
2-4-1	Team assemble	mechanical : cutting / severing not paying attention when manufacturing frame	Serious Unlikely	Medium	warning label(s)	Serious Unlikely	Medium	TBD [3/19/2021] juan
2-4-2	Team assemble	mechanical : pinch point body part too close to system	Serious Likely	High	warning label(s)	Serious Unlikely	Medium	TBD [3/19/2021] caleb
2-4-3	Team assemble	mechanical : stabbing / puncture not paying attention	Serious Unlikely	Medium	warning label(s)	Serious Unlikely	Medium	TBD [3/19/2021] ryan
2-5-1	Team storage	mechanical : product instability incorrect assembly	Moderate Remote	Negligible	special tools or fixtures	Moderate Unlikely	Low	TBD [3/19/2021] nick

Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods /Control System	Final Assessment		Status / Responsible /Comments /Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
2-6	Team misuse	<None>						
3-1-1	End User (Student) General Use	mechanical : drawing-in / trapping / entanglement body part too close to system	Serious Likely	High	fixed enclosures / barriers	Serious Unlikely	Medium	TBD
3-1-2	End User (Student) General Use	mechanical : pinch point body part too close to system	Serious Likely	High	fixed enclosures / barriers	Serious Unlikely	Medium	TBD
3-1-3	End User (Student) General Use	electrical / electronic : improper wiring incorrect assembly	Serious Unlikely	Medium	standard procedures	Serious Unlikely	Medium	TBD
3-1-4	End User (Student) General Use	electrical / electronic : overloading controller error	Serious Unlikely	Medium	instruction manuals	Serious Unlikely	Medium	TBD
3-1-5	End User (Student) General Use	electrical / electronic : unexpected start up / motion controller error	Serious Unlikely	Medium	standard procedures	Serious Unlikely	Medium	TBD
3-1-6	End User (Student) General Use	slips / trips / falls : impact to / with horseplay around system	Serious Unlikely	Medium	supervision	Serious Unlikely	Medium	TBD
3-1-7	End User (Student) General Use	slips / trips / falls : instability motor error	Serious Unlikely	Medium	instruction manuals	Serious Remote	Low	TBD
3-1-8	End User (Student) General Use	slips / trips / falls : falling material / object incorrect assembly/motor error	Serious Unlikely	Medium	standard procedures	Serious Unlikely	Medium	TBD

Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods /Control System	Final Assessment		Status / Responsible /Comments /Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
3-1-9	End User (Student) General Use	ergonomics / human factors : lifting / bending / twisting reposition system	Moderate Unlikely	Low	cart	Moderate Unlikely	Low	TBD
3-1-10	End User (Student) General Use	noise / vibration : noise / sound levels > 80 dBA motor error	Moderate Unlikely	Low	standard procedures	Moderate Unlikely	Low	TBD
3-1-11	End User (Student) General Use	noise / vibration : product / equipment damage falling off table	Moderate Unlikely	Low	standard procedures	Moderate Unlikely	Low	TBD
3-1-12	End User (Student) General Use	fluid / pressure : fluid leakage / ejection to much lubricant	Moderate Unlikely	Low	standard procedures	Moderate Unlikely	Low	TBD
3-2-1	End User (Student) Misuse	mechanical : Projectile motor error	Serious Unlikely	Medium	fixed enclosures / barriers	Serious Unlikely	Medium	TBD
3-2-2	End User (Student) Misuse	electrical / electronic : water / wet locations water bottle near system	Serious Unlikely	Medium	supervision	Serious Unlikely	Medium	TBD
3-2-3	End User (Student) Misuse	electrical / electronic : unexpected start up / motion motor error	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD
3-2-4	End User (Student) Misuse	electrical / electronic : Tamper with components lack of knowledge	Serious Remote	Low	Display Manual	Serious Unlikely	Medium	TBD
4-1-1	passer-by / non-user walk near	mechanical : Projectile motor error	Moderate Unlikely	Low	fixed enclosures / barriers	Moderate Unlikely	Low	TBD

Item Id	User / Task	Hazard / Failure Mode	Initial Assessment		Risk Reduction Methods / Control System	Final Assessment		Status / Responsible / Comments / Reference
			Severity Probability	Risk Level		Severity Probability	Risk Level	
4-2-1	passer-by / non-user misuse	electrical / electronic : improper wiring assembly error	Serious Unlikely	Medium	standard procedures	Serious Unlikely	Medium	TBD
4-3-1	passer-by / non-user observe / watch	noise / vibration : noise / sound levels > 80 dBA motor error	Moderate Unlikely	Low	E-stop control	Moderate Unlikely	Low	TBD
5-1-1	Repair Technician General Repair Task	mechanical : pinch point body part too close to system while running	Serious Unlikely	Medium	fixed enclosures / barriers	Serious Unlikely	Medium	TBD
5-1-2	Repair Technician General Repair Task	electrical / electronic : improper wiring assembly error	Serious Unlikely	Medium	standard procedures	Serious Unlikely	Medium	TBD
5-1-3	Repair Technician General Repair Task	electrical / electronic : overloading motor error	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD
5-1-4	Repair Technician General Repair Task	electrical / electronic : overvoltage / overcurrent controller/ motor error	Serious Unlikely	Medium	E-stop control	Serious Unlikely	Medium	TBD
5-1-5	Repair Technician General Repair Task	electrical / electronic : power supply interruption trip over cord	Serious Unlikely	Medium	warning label(s)	Serious Unlikely	Medium	TBD
5-1-6	Repair Technician General Repair Task	electrical / electronic : Tamper with components lack of knowledge	Serious Remote	Low	instruction manuals	Serious Unlikely	Medium	TBD
5-1-7	Repair Technician General Repair Task	ergonomics / human factors : lifting / bending / twisting repositioning for demonstration	Moderate Unlikely	Low	cart	Moderate Unlikely	Low	TBD

A.17 Electrical Schematics/Wiring Diagrams



A.18 Annotated Microcontroller Code

```
'''
@file      controller.py
@brief     This file contains the controller class used to institute PID
control on the CNC Feed Drive.
@details   This file contains a controller object that acts as a closed loop
system.

@author    Ryan Funchess
@date      June 1, 2021
'''

import pyb
import utime
import math
import shares
pi = math.pi

class controller:
    '''
    @brief   This Class implements a closed loop control algorithm

    #option for P, PI, or PD control
    '''

    def __init__(self, Kp, Ki, Kd, posref, mot, enc, LimitSwitch, interval):
        '''
        @brief   Makes controller object
        @param Kp   Float that designates the position controller gain for the
control system (V/rpm)
        @param Ki   Float that designates the position controller gain for the
control system (V/rpm)
        @param Kd   Float that designates the position controller gain for the
control system (V/rpm)
        @param posref   Float that designates reference velocity of motor
(rpm)
        @param mot    Motor object that rotates output shaft proportional to
the duty cycle of the motor
        @param enc    Encoder object that is used to track the position of the
output shaft
        @param cloop   Closed loop object that is used to determine the
voltage to apply to the motor given the current speed of the motor
        @param LimitSwitch   Float that designates the position controller
gain for the control system (V/rpm)
        @param interval   Float that designates the position controller gain
for the control system (V/rpm)
        '''
```

```

control    ##Float that designates the proportional controller gain for the
system    system (V/rpm)
        self.Kp=Kp #passed Kp (V/cm)

system    ##Float that designates the integral controller for the control
system    system
        self.Ki=Ki #passed Ki (V/cm*s)

system    ##Float that designates the derivative controller for the control
system    system
        self.Kd=Kd #passed Kd (V*s/cm)

        ##Float that designates reference position of system (rpm)
        self.posref=posref #designated reference position in cm

cycle of the motor    ##Motor object that rotates output shaft proportional to the duty
        self.mot=mot

shaft    ##Encoder object that is used to track the position of the output
        self.enc=enc

        ##Limit switches and lid switches combined
        self.LimitSwitch=LimitSwitch

        ##Interval at which controller runs in ms
        self.interval=interval

        ##Sum of the error using a Riemann Sum for the integral control
        self.e_sum=0 #MAKE SURE TO RESET THIS WHENEVER POSREF CHANGES

        ##Previous error to compute dx/dt
        self.prev_error=0 #use for D control

def startup(self): #runs first, zeros table to limit switch
    #bring table to limit switch and then stop
    #zero encoder shares.clearpos=1
    #while the left limit switch is not reading true:
    #    set motor duty cycle to -50
    #set motor duty to 0 after breaks through loop
    #zero the encoder
    #await
    '''
    @brief This method zeroes the actuating system upon startup
    @details Upon startup, the run of this function causes the ballnut to
move        in the negative direction until a limit switch is encountered, at
        which point the

```



```

encoder is zeroed. To untrigger the switch, the system then actuates
in open loop
    to a position of about 1cm in the positive direction.
'''

    while(not(self.LimitSwitch.read())): #until a limit switch is
contacted. zero to thrust bearing side
        self.mot.set_duty(-31) #move in negative direction

    self.mot.set_duty(0) #stop motor
    self.enc.update() #update encoder
    shares.clearpos=1 #tell encoder to zero on next update
    self.enc.update() #update encoder

    while(self.enc.get_linearpos()<1): #actuate to 1cm position in open
loop
        #print(self.enc.get_linearpos())
        self.mot.set_duty(29) #move in positive direction
        self.enc.update() #update the encoder in order to get linear
position

        self.mot.set_duty(0) #stop motor


def run(self):
    #write to UART
    '''
    @brief This method runs the closed loop control system
    @details This method runs a single iteration of a closed loop
controller. Encoders update, the controller
        denotes the voltage to output to the motor, and the duty cycle is set
as a percentage to the motor. Data is then returned
        as a tuple of the form [position[cm], velocity[cm/s], angular
velocity[rpm],motor duty cycle [%]]. If a limit switch is engaged
        due to overshoot of the system, the motor is disabled.
    '''
    #pseudocode
    #if either limit switch isnt engaged
        #update encoder
        #get position and speed
        #obtain voltage to apply to motor(from control system)
        #set motor duty cycle
        #return time, position, velocity (write to uart in main method)

    #else if the limit switches are engaged
        #disable motor
        #set duty cycle to 0

    if(not(self.LimitSwitch.read())): #if either limit switch is not
engaged

```

```

        self.enc.update() #updates encoder to find current delta
        posact=self.enc.get_linearpos() #linear position[cm]
        omega=self.enc.get_speed() #obtain current velocity based on
delta and interval time (rpm)
        velocity=self.enc.get_linearvel() #linear speed in cm/s
        level=self.controlVoltagePID(posact)/12*100 #obtain level to
apply to motor, convert V to %max voltage. Obtained from required
voltage/12=fractional percent of 12V, then multiply by 100
        self.mot.set_duty(level) #set duty cycle on motor input voltage
based on output from cloop
        return [posact,velocity,omega,level] #return data

    else: #if limit switch engaged
        self.mot.disable() #disable motor
        self.mot.set_duty(0) #set duty cycle to 0

def set_pos(self,newpos):
    """
    @brief This function is a setter function for reference position.
Integral buildup is also reset.
    @param newpos New reference position upon which error is calculated
in cm.
    """
    self.posref=newpos #set posref to newpos
    self.e_sum=0 #reset PI

def getPosRef(self):
    """
    @brief This function returns the current value of posref held in this
class.
    """
    return self.posref

def controlVoltageP(self,pos):
    """
    @brief This function implements a P only control system.
    @param pos Current position in cm measured by encoders
    @return Vp Requested output voltage to the motor
    """
    #convert angle to distance with pitch
    Vp=self.Kp*(self.posref-pos) # positive delta=positive
voltage=+change in position
    return Vp

def controlVoltagePI(self,pos):
    """
    @brief This function implements a PI control system.
    @param pos Current position in cm measured by encoders
    @return Vp Requested output voltage to the motor
    """
    error=self.posref-pos #calculated error

```

```

        self.e_sum=self.e_sum+error*(self.interval/1000) #added error to sum
of error multiplied by interval in s, numerical integration

        Vp=self.Kp*error+self.Ki*self.e_sum #requested voltage output with
implemented P and I
        return Vp

    def controlVoltagePID(self,pos):
        '''
        @brief This function implements a P only control system.
        @param pos Current position in cm measured by encoders
        @return Vp Requested output voltage to the motor
        '''
        error=self.posref-pos #calculated error

        self.e_sum=self.e_sum+error*(self.interval/1000) #added error to sum
of error multiplied by interval in s, numerical integration

        Vp=self.Kp*error+self.Ki*self.e_sum+self.Kd*(error-
self.prev_error)/(self.interval/1000) #implemented PID control
        self.prev_error=error #store previous error for D control

        return Vp

    def openlooprun(self,oldduty):
        '''
        @brief This function runs the system in open loop for system
characterization.
        @details This function runs the system at the applied duty cycle for
20cm to allow
        students to characterize the system, experimentally find Kp, etc.
        @param pos Current position in cm measured by encoders
        @return Vp Requested output voltage to the motor
        '''
        if(self.enc.get_linearpos()<20 and not(self.LimitSwitch.read())):
#run in open loop until at 20cm or limit switch is hit
            self.mot.set_duty(oldduty) #move in positive direction
            self.enc.update() #update the encoder in order to get linear
position
            posact=self.enc.get_linearpos() #linear position[cm]
            omega=self.enc.get_speed() #obtain current velocity based on
delta and interval time (rpm)
            velocity=self.enc.get_linearvel() #linear speed in cm/s

            return [posact,velocity,omega,oldduty] #return data
        else: #otherwise its stop time
            self.mot.set_duty(0) #stop motor
            self.mot.disable() #disable motor
            self.enc.update() #update the encoder in order to get linear
position
            posact=self.enc.get_linearpos() #linear position[cm]
            omega=self.enc.get_speed() #obtain current velocity based on
delta and interval time (rpm)

```

```

velocity=self.enc.get_linearvel() #linear speed in cm/s
return [posact,velocity,omega,olduty] #return data


'''
@file EncoderLab6.py

@brief This file allows for use of the encoder.

'''

import pyb
import shares

class Encoder:
    '''
    @brief      This class interacts with encoder hardware and updates
encoder position on the MCU
    @param position    Integer position of encoder from counter, updated by
the update() method
    @param zeroer      Integer that is subtracted from delta in order to zero
the position of the encoder after user input, updated at end of update()
    @param pitch Float that represents the pitch of the ballscrew in cm. Our
ballscrew has a pitch of 5mm.
    '''
    ##Integer position of encoder from counter, updated by the update()
method
    position=0

    ##Integer that is subtracted from delta in order to zero the position of
the encoder after user input, updated at end of update()
    zeroer=0

    ##Pitch in cm/rot
    pitch=0.5

    def __init__(self,tim,period,interval,CPR):

```

```

'''
    @brief      Creates an encoder object
    @param tim   Timer object used with two channels connected to
encoder phase
    @param period   Period for timer
'''

##Timer object used with two channels connected to encoder phase
self.tim=tim

##Period for timer
self.period=period

self.interval=interval

self.CPR=CPR

def update(self):
'''
    @brief      Updates encoder position
'''

    curread=self.tim.counter() #obtain current reading on encoder

    #Compute delta from new reading and current position, zeroer is
    additionally subtracted
    delta=self.get_delta(self.position+self.zeroer,curread)

    if(abs(delta)>=(self.period/2)): #bad delta if greater than or equal
to half of the period
        if(delta>0): #positive bad delta
            delta=delta-self.period #correct by subtracting period
        else: #negative bad delta
            delta=delta+self.period #correct by adding period

    self.delta=delta
    # Look at shares.py, which contains shared variables between user
    interface and this class.
    # If clearpos==1, user desires the position reading to be zeroed.
    if(shares.clearpos==1):
        self.zeroer+=self.get_position() #Current position is added to
zeroer, this is subtracted from delta to decrease/increase future deltas
        self.set_position(0) #The current position is set to 0
        shares.clearpos=0 #The reset variable is reset

    newpos=self.position+delta #new position as calculated as the old
    position plus the corrected delta
    self.set_position(newpos) #new position is set

    #The current delta and position readings are put into shares.py for
    interaction with the user interface.

```

```

        shares.delta=delta
        shares.position=newpos

    def get_position(self):
        """
        @brief      Returns encoder position in ticks. MUST BE RUN AFTER
UPDATE.
        """
        return self.position

    def get_positionDeg(self):
        """
        @brief      Returns encoder position in degrees. MUST BE RUN AFTER
UPDATE.
        """
        conv=self.CPR*4/360 #yields ticks/deg
        posdeg=self.position/conv
        return posdeg

    def get_positionRot(self):
        """
        @brief      Returns encoder position in rotations. MUST BE RUN AFTER
UPDATE.
        """
        conv=1/360
        return(self.get_positionDeg()*conv)

    def set_position(self,newpos):
        """
        @brief      Sets encoder position in ticks
        """
        self.position=newpos

    def get_delta(self,position,curread):
        """
        @brief      Returns the difference between two values that are
passed to this method
        """
        delta=curread-position
        self.delta=delta
        return delta

    def get_speed(self): #MAKE SURE TO RUN THIS AFTER UPDATE
        """
        @brief Return speed in rpm. MUST BE RUN AFTER UPDATE.
        """
        conv=self.CPR*4 #yields ticks/rev
        dx=(self.delta)/conv #revolutions since last check
        dt=self.interval/(60000) #interval converted from ms to minute
        speed=dx/dt #calculate speed from change in position over change in
time

```

```

    return speed

def get_linearpos(self): #MAKE SURE TO RUN THIS AFTER UPDATE
    '''
    @brief Returns linear position in cm. MUST BE RUN AFTER UPDATE.
    '''
    #returns linear position of mass in cm
    rot=self.get_positionRot() #rotational position
    dist=self.pitch*rot
    return dist

def get_linearvel(self): #MAKE SURE TO RUN THIS AFTER UPDATE
    '''
    @brief Returns linear velocity in cm/s.MUST BE RUN AFTER UPDATE.
    '''
    #returns linear velocity in cm/s
    rpm=self.get_speed() #angular velocity in rpm
    vel=rpm/60*self.pitch #rpm*60=rot/s, rot/s*pitch(dist/rot)=dist/s
    return vel

if __name__=="__main__":

    # period=0xffff #Period of timer
    # interval=30 #interval in ms
    # tim=pyb.Timer(4) #Declare timer 4
    # tim.init(prescaler=0,period=0xFFFF) #Initialize timer 4 with prescale
and period
    # tim.channel(1,pin=pyb.Pin.cpu.B6,mode=pyb.Timer.ENC_AB) #channel 1 of
timer 4 to pin B6
    # tim.channel(2,pin=pyb.Pin.cpu.B7,mode=pyb.Timer.ENC_AB) #channel 2 of
timer 4 to pin B7

    # CPR=25.9 #CPR of motor output shaft, quad encoder
    # enc=Encoder(tim,period,interval,CPR) #declare encoder object, pass
timer,period, interval

    period=0xffff #Period of timer
    interval=30 #interval in ms
    tim=pyb.Timer(5) #Declare timer 4
    tim.init(prescaler=0,period=0xFFFF) #Initialize timer 4 with prescale and
period
    tim.channel(1,pin=pyb.Pin.cpu.A0,mode=pyb.Timer.ENC_AB) #channel 3 of
timer 5 to pin A2
    tim.channel(2,pin=pyb.Pin.cpu.A1,mode=pyb.Timer.ENC_AB) #channel 4 of
timer 5 to pin A3

    CPR=25.9 #CPR of motor output shaft, quad encoder
    enc=Encoder(tim,period,interval,CPR) #declare encoder object, pass
timer,period, interval

```



```

while True:
    enc.update()
    print(enc.get_positionDeg())

'''
@file      LimitDriver.py
@brief     This file contains the hardware driver for the limit switches.
@author    Ryan Funchess
@date      June 1, 2021
'''

import pyb
import utime
from pyb import Pin
from pyb import ADC

class LimitSwitch:
    '''
    @brief  This Class implements a limit switch'''

    # def __init__(self,sensor):
    #     '''
    #     @brief  Makes limit switch object
    #     @param sensor Pin object connected to limit switch
    #     '''
    #     self.sensor=sensor
    #     self.sensor.init(mode=Pin.ANALOG)
    #     self.reader=ADC(self.sensor)

    def __init__(self,sensor):
        '''
        @brief  Makes limit switch object
        @param sensor Pin object connected to limit switch
        '''
        self.sensor=sensor

    def read(self):
        '''

```

```

        @brief This function reads from the limit switch and returns True if
engaged.
        @details Returns True if limit switch is engaged and False if no
obstruction is detected.
        '''
        reading=self.sensor.value() #obtains 1 if high, 0 if low
        if reading==1: #if high then circuit is open
            return True
        return False

if __name__=="__main__":

    sensorPin=Pin(Pin.cpu.A0,Pin.IN,Pin.PULL_UP)
    sensor=LimitSwitch(sensorPin)

    while True:
        sensor.read()
'''
@file MotorDriver.py
This class has been designed to work with physical motor hardware. Pulse
Width Modulation (PWM) is used
in order to drive motor based on a designated user input. The motor is also
able to be enabled and disabled
by setting the Enable pin high or low respectively.
'''

import pyb
import utime

class MotorDriver:
    '''
    @brief This Class implements a motor'''

    def __init__(self,nSLEEP_Pin,IN1_Pin,IN2_Pin,timer):
        '''
        @brief Makes motor object.
        @details Constructs motor object. Also sets saturation limit to 50%,
as we are running a 12V motor from a 24V variable source.
        @param nSLEEP_Pin Enable pin
        @param IN1_Pin In 1 Pin
        @param IN2_Pin In 2 Pin
        @param timer Timer that works with PWM for IN1_Pin and IN2_Pin
        '''
        # print('Creating a motor driver')
        ##Enable Pin
        self.nSLEEP_Pin=nSLEEP_Pin
        ##In 1 Pin
        self.IN1_Pin=IN1_Pin
        ##In 2 Pin
        self.IN2_Pin=IN2_Pin
        ##Timer that works with PWM for IN1_Pin and IN2_Pin
        self.timer=timer

```

```

self.tch1=self.timer.channel(1,pyb.Timer.PWM,pin=self.IN1_Pin)
self.tch2=self.timer.channel(2,pyb.Timer.PWM,pin=self.IN2_Pin)

##Saturation limit for motor, cannot go above this duty cycle
self.satlimit=50

def enable(self):
    '''
    @brief This function enables the motor to run by setting nSLEEP high.
    '''
    self.nSLEEP_Pin.high()
    # print('Enabling motor')

def disable(self):
    '''
    @brief This function disables the motor by setting nSLEEP low.
    '''
    self.nSLEEP_Pin.low()
    # print('Disabling motor')

def set_duty(self,duty):
    '''
    @brief This function sets the duty cycle on the PWM for the motor,
    which by
    increasing average voltage applied to the motor, will increase motor
    speed. This function checks
    to make sure that the input is valid, and then sets the pins to the
    proper value to ensure the right
    duty cycle in the correct direction.
    '''
    if(not(self.is_float(duty))): #check to make sure that the passed
value is a float
        pass
    else:
        if(abs(duty)>self.satlimit): #saturation, so if a value over 50%
is requested, 50% is maximum possible that it will return
            duty=self.satlimit*abs(duty)/duty #abs(duty)/duty puts
voltage in direction of originally requested duty
            if(duty>=0): #rotate forward
                self.tch1.pulse_width_percent(duty) #set forward pin to
proper duty cycle
                self.tch2.pulse_width_percent(0) #keep other pin low
            else: #reverse direction
                self.tch1.pulse_width_percent(0) #set forward pin low
                self.tch2.pulse_width_percent(-1*duty) #set reverse pin to
magnitude of passed value. Multiplying by negative corrects sign.

        ##Duty cycle which controls average voltage applied to motor,
controlling motor output speed
        self.duty=duty

```

```

        # print(duty)
        # print('Running with new duty cycle of '+str(duty)+' %')

def is_float(self,n):
    """
    @brief This function is used to ensure that a passed string is able
    to be converted
    to a float. The general outline of this method was obtained from the
    following link:
    https://note.nkmmk.me/en/python-check-int-
    float/#:~:text=Check%20if%20a%20number%20is%20integer%20or%20decimal,3%20Chec
    k%20if%20numeric%20string%20is%20integer.%20'''
    try:
        float(n)
    except ValueError:
        return False #if the attempt to convert the value to float has an
        error, return false immediately
    else:
        n=float(n)
        return isinstance(n,float) #returns boolean determining if value
        is float

if __name__=='__main__':
    #PB4 and PB5
    pinA10=pyb.Pin(pyb.Pin.cpu.A10,pyb.Pin.OUT_PP)
    pinB4=pyb.Pin(pyb.Pin.cpu.B4)
    tim3=pyb.Timer(3,freq=20000)
    t3ch1=tim3.channel(1,pyb.Timer.PWM,pin=pinB4)
    pinB5=pyb.Pin(pyb.Pin.cpu.B5)
    t3ch2=tim3.channel(2,pyb.Timer.PWM,pin=pinB5)

    moe1=MotorDriver(pinA10,pinB4,pinB5,tim3)
    moe1.enable()

    """
main.py

Test code for closed loop with P, PI, PID and serial communication
Step input from serial communication
Test Code 5 will do open loop test and safety
"""

from LimitDriver import LimitSwitch
from MotorDriver import MotorDriver
from EncoderLab6 import Encoder
from controller import controller
import pyb
from pyb import Pin
import utime
import shares

```

```

import array

#initialize limit switch, wait for lid to be put on
#sensor

sensorPin=Pin(Pin.cpu.A6,Pin.IN,Pin.PULL_UP)
sensor=LimitSwitch(sensorPin)

while(sensor.read()): #true while the lid is off, will not do anything until
lid is placed
    pass

while True:
    myuart=pyb.UART(2) #initialize UART for communication with computer

    #waiting for entire string of format Kp, Ki, Kd, posref
    n=0 #loop control variable
    while(n==0): #runs while n==0, changed from 0 when Kp received
        if myuart.any() != 0: #if there's anything from the computer
            frontendreq = myuart.read().decode('ascii') #read the line
            requests=frontendreq.split(',') #split input string on space
            Kp=requests[0] #first entry is Kp
            Kp=float(Kp) #convert to float
            Ki=requests[1] #second entry is Ki
            Ki=float(Ki) #convert to float
            Kd=requests[2] #third entry is Kd
            Kd=float(Kd) #convert to float
            posref=requests[3] #first reference position
            posref=float(posref) #convert to float
            posref2=requests[4] #second references position
            posref2=float(posref2) #convert to float
            posref3=requests[5] #third reference position
            posref3=float(posref3) #convert to float
            typeofloop=requests[6] #0 for open loop, 1 for closed loop
            typeofloop=int(typeofloop) #convert to int
            openduty=requests[7] #open loop duty cycle
            openduty=float(openduty) #convert to float

            positions=[posref,posref2,posref3] #put desired positions in list
to be iterated

            n=1 #break out of loop

    #Motor
    pinA10=pyb.Pin(pyb.Pin.cpu.A10,pyb.Pin.OUT_PP) #enable Pin
    pinB4=pyb.Pin(pyb.Pin.cpu.B4) #channel 1 for PWM on motor
    tim3=pyb.Timer(3,freq=20000) #PWM uses timer 3
    t3ch1=tim3.channel(1,pyb.Timer.PWM,pin=pinB4) #timer 3 channel 1
    pinB5=pyb.Pin(pyb.Pin.cpu.B5) #channel 2 for pwm on motor
    t3ch2=tim3.channel(2,pyb.Timer.PWM,pin=pinB5) #timer 3 channel 2

```

```

moel=MotorDriver(pinA10,pinB4,pinB5,tim3) #create motor object
moel.enable() #enable the motor

#encoder
period=0xffff #Period of timer
interval=30 #interval in ms
tim=pyb.Timer(5) #Declare timer 4
tim.init(prescaler=0,period=0xFFFF) #Initialize timer 4 with prescale and
period
tim.channel(1,pin=pyb.Pin.cpu.A0,mode=pyb.Timer.ENC_AB) #channel 3 of
timer 5 to pin A2
tim.channel(2,pin=pyb.Pin.cpu.A1,mode=pyb.Timer.ENC_AB) #channel 4 of
timer 5 to pin A3

CPR=25.9 #CPR of motor output shaft, quad encoder
enc=Encoder(tim,period,interval,CPR) #declare encoder object, pass
timer,period, interval

#Controller, pass gains and initial reference position and hardware
drivers
cont=controller(Kp,Ki,Kd,posref,moel,enc,sensor,interval)

#startup/zero the system
cont.startup()

if(typeofloop==1): #closed loop
    ## The integer timestamp for the first iteration
    start_time = utime.ticks_ms()
    # print('Start time: '+str(self.start_time))

    ## The integer "timestamp" for when the next run should be
    next_time = utime.ticks_add(start_time, interval)

    #Data collection
    TimeData=array.array('i',[]) #list to store time values in ms
    SpeedData=array.array('f',[])#list to store velocity data in rpm
    PosData=array.array('f',[]) #list to store position values in cm

    curr_time = utime.ticks_ms() #current time

    posi=0 #indicates which of the three positions the controller is
referenced to
    stop=False #tells the loop when to stop
    while(not(stop)): #runs for 15 seconds
        curr_time = utime.ticks_ms() #current time
        # if myuart.any() != 0: #if there's anything from the computer
        #     newpos = myuart.read().decode('ascii') #read the line
        #     newpos=float(newpos)
        #     cont.set_pos(newpos)
        if utime.ticks_diff(curr_time,next_time)>0: #runs the task if
current time is past the next time

```

```

        data=cont.run() #run the control loop
        PosData=data[0] #current position is passed from controller
        SpeedData=data[1]
        TimeData=curr_time-start_time #change in time from beginning
to now

myuart.write('{:},{:},{:}\r\n'.format(TimeData,PosData,SpeedData)) #write
data over (real time data collection)
        if(data[2]<=8 and abs(data[0]-positions[posi])<=0.05): #goes
to next position at steady state
            #steady state defined as requested duty cycle<8, error in
position less than 0.01cm
            posi=posi+1 #iterate to next
            if(posi==3): #when all reference positions have been
iterated thru
                stop=True #stop the loop
            else:
                cont.set_pos(positions[posi]) #set reference position
                next_time = utime.ticks_add(next_time, interval) #set the
next time for the task to actually run

        moel.disable() #disable motor

        myuart.write('DION\r\n') #passes the terminator to tell the frontend
that data collection is done

elif(typeofloop==0): #open loop
    ## The integer timestamp for the first iteration
    start_time = utime.ticks_ms()
    # print('Start time:'+str(self.start_time))

    ## The integer "timestamp" for when the next run should be
    next_time = utime.ticks_add(start_time, interval)

    #Data collection
    TimeData=array.array('i',[]) #list to store time values in ms
    SpeedData=array.array('f',[])#list to store velocity data in rpm
    PosData=array.array('f',[]) #list to store position values in cm

    curr_time = utime.ticks_ms() #current time

    stop=False #tells the loop when to stop
    while(not(stop)): #
        curr_time = utime.ticks_ms() #current time
        if utime.ticks_diff(curr_time,next_time)>0: #runs the task if
current time is past the next time
            data=cont.openlooprun(openduty) #run the open loop controller
            PosData=data[0] #current position is passed from controller

```

```

        SpeedData=data[1] #current speed passed from controller
        TimeData=curr_time-start_time #change in time from beginning
to now

myuart.write('{:},{:},{:}\r\n'.format(TimeData,PosData,SpeedData)) #write
data over (real time data collection)
        if(data[0]>19): #stops after travelling 19cm
            moel.disable() #disable motor
            stop=True #stop the loop

            next_time = utime.ticks_add(next_time, interval) #set the
next time for the task to actually run

        moel.disable() #disable motor


        myuart.write('DION\r\n') #passes the terminator to tell the frontend
that data collection is done

```


A.19 Test Procedures

Test Procedure - Table Travel

Test Name: Table Travel

Purpose: *To verify that the "Table" travels 30 cm in a linear fashion.*

Scope: This test will verify that the table is able to travel 30 cm along the ballscrew's longitudinal axis per our sponsor's product specification; note that the table must be do so without impedance by misalignment or other failure.

Equipment:

- Assembled feed drive system
 - o Table
 - o Ballscrew/Ballnut/Ball and Thrust Bearing
 - o Linear Rails/Bearings
 - o Frame
- Surface for system to be placed on
- Measuring tool (Dial calipers)
- Tape
- Marker

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <70lbs)

PPE Requirements:

- Safety Goggles
- Face Mask
- Do not wear jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Ensure mechanical fasteners are correctly tightened.
- 2) Wear required PPE (safety glasses, tie hair back, and put away jewelry)
- 3) Add tape to either end of one linear rail, use a marker to draw a dash on the top surfaces of the tape, and use calipers to space the dashes 30 cm apart. Note any points of binding.
- 4) Manually turn the ballscrew to verify that it produces linear motion of the table.
- 5) Mark the table's center position by placing tape on its side and using a marker to exactly scribe the table's center on the piece of tape.
- 6) Set the table's initial center position to be aligned with the dash on either piece of tape.
- 7) Once the table begins its motion, run it until its center position is in-line with the dash of the other piece of tape. This will have been 30cm of table travel distance.

Results:

Pass Criteria: Table travels 30cm linearly without binding.

Fail Criteria: Table is not able to reach 30cm of travel.

of Samples: Run test three times.

Test Procedure - Microcontroller Communication

Test Name: Microcontroller Communication (Motor, Encoder, and Limit Switches)

Purpose: *The purpose of this test is to ensure that we can accurately control the voltage outputs on the Nucleo Microcontroller and ensure that the Nucleo can accurately read from the motor, encoder and limit switch pins. Additionally, we must ensure serial communication between the computer and the microcontroller.*

Scope: The scope of this test is to ensure that the Nucleo Microcontroller is fully functional, able to accurately read from input pins and output predictable voltage. Verification is required of the USB connection, motor driver pin outputs, encoder pin outputs and inputs, and ADC readers for the limit switches.

Equipment:

- Jumper cables
- Multimeter
- Nucleo Microcontroller
- Motor Driver Breakout Board
- USB 2.0 cable

Hazards:

- Static discharge from hands to microcontroller
- Short circuit across board
- Electric shock to users

PPE Requirements:

- Avoid contact with electrical components with bare hands.
- DO NOT plug in any components without verifying wiring with teammates.

Facility: Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Connect the Motor Driver hardware to the Nucleo by stacking the breakout board onto the correct pins on the Nucleo.

2) Power on the Nucleo by connecting the computer to the Nucleo's local USB port with the USB 2.0 cable. Ensure that the jumper cable controlling Nucleo power on the board is connected to U5V, rather than E5V.

3) Ensure access to the Nucleo via PuTTY. Print a few lines to the PuTTY terminal to ensure board functionality.

Serial Communication

4) Push main.py from the "Test Folder" to the Nucleo storage by pushing via the Command Prompt.

5) Power off the Nucleo by unplugging the USB cord. Plug the Nucleo back in so that main.py runs on startup.

6) Test Serial Communication via UART to the Nucleo by running the UIFrontend.py file on the PC side. The Nucleo should return values via the Serial Port at the completion of UIFrontend.py.

Encoder Pins

7) Using a multimeter, ensure a 5V difference between the native board 5V pin and ground pin. This will be used to power the encoders and limit switches.

8) Set Pins B6 and B7 to read ADC input and short the 5V pin to each of the pins. Read from the ADC on each pin and verify a value of around 4095 (for a 12-bit ADC). These pins will be used to read data from the encoders.

Motor Pins

9) Set Pin A15 to be an output pin. Set the pin high and then low and use a multimeter to verify that the pin produces a voltage of 3.3V and 0V, respectively.

10) Initialize Pins B4 and B5 and initialize to do Pulse Width Modulation (PWM) from those pins. Verify that linearly varying duty cycle between 0 and 100 linearly varies the pin output voltage with a multimeter. Additionally, measure M- and M+ on the motor driver board and ensure that the voltage matches Pins B4 and B5 respectively.

Limit Switches

11) Set Pins A0 and A1 to read ADC input and short the 5V pin to each of the pins. Read from the ADC on each pin and verify a value of around 4095 (for a 12-bit ADC). These pins will be used to read from the proximity sensors.

Results: The test is passed if the Nucleo can successfully communicate with the computer via serial port, match pin outs with expected values, and read an external input via ADC on all relevant pins.

Test Procedure - Load

Test Name: Load Requirement

Purpose: *To verify that the Table, Ballscrew, and Linear Rails can withstand 5 kg load without deflection and still travel linearly.*

Scope: This test will verify that our system adequately handles the mass plates that are loaded to the top of the feed drive table. We define “adequate” handling to be no deflection in the rails nor ballscrew, and continued linear motion of the table for rotations of the ballscrew.

Equipment:

- Assembled feed drive system
- Mass plates
- Surface for system to be placed on
- Measuring tool (Dial calipers)
- String
- Tape

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <70lbs)
- Pinch points when attaching mass plates to the top of table

PPE Requirements:

- Safety Goggles
- Face Mask
- Do not wear jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Ensure mechanical fasteners are correctly tightened.
- 2) Wear required PPE (safety glasses, tie hair back, and put away jewelry)
- 3) Fix string to either end of the linear rails and ballscrew, such that it falls in line with the top vertex of the rail/ballscrew.
- 4) Place masses atop the table and secure them using fasteners.
- 5) Observe deflection in the rails/ballscrew by using dial calipers to measure any difference between the fixed string and the top vertex of the supports (rail+ballscrew).
- 6) Record any deflection and compare to 0.05 in maximum allowed deflection.
- 7) Manually turn the ballscrew and observe the linear motion of the table. Check for any points of binding.

Results: Pass Criteria, Fail Criteria, Number of samples to test

Run test procedure with the maximum plate loading of 5kg.

Pass Criteria: Deflection is under 0.05 in and table moves linearly without binding.

Test Procedure - Table Speed

Test Name: Table Speed

Purpose: *To verify that the "table" travels at a top speed of 10 cm/s.*

Scope: This test will verify that the table is able to travel at a top speed of 10 cm/s along the ballscrew's longitudinal axis; note that the encoder, microcontroller, limit switch, and ballscrew pitch test procedures must be completed prior to this test.

Equipment:

- Assembled feed drive system
- Surface for system to be placed on

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <70lbs)
- System failure (mechanical or electrical)

PPE Requirements:

- Safety Goggles
- Face Mask
- Do not wear jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Ensure mechanical fasteners are correctly tightened.
- 2) Verify that the encoder, microcontroller, limit switch, and ballscrew pitch test procedures have been completed prior to this test.
- 3) Run test procedure with the maximum plate loading of 5kg.

- 4) Run table at maximum system speed from one end of the linear axis to the other.
- 5) Observe collected speed data and verify that a maximum table speed of 10 cm/s was reached in the duration of the run. NOTE: This table speed corresponds to a motor output speed of 1800 rpm.
- 6) Return table to the start of the linear axis at maximum speed and verify the output data indicates a top speed of 10 cm/s.
- 7) Run test procedure

Results: Pass Criteria, Fail Criteria, Number of samples to test

Pass Criteria: System data output indicates the table traveled at a maximum speed of 10 cm/s in a run across the linear axis.

Fail Criteria: The table does not reach 10 cm/s in the duration of its run.

Number of Samples: 2

Test Date(s): Following completion of the encoder, microcontroller, limit switch, and ballscrew pitch test procedures.

Test Procedure - Mobility

Test Name: Mobility

Purpose: *To verify that the entire feed drive system weighs less than or equal to 50 lb.*

Scope: This test will verify that our system is portable by maintaining an overall weight of less than or equal to 50 lb and can be carried by one person.

Equipment:

- Assembled feed drive system
- Weight Scale

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <70lbs)
- Dropping the system when transferring it to the weight scale

PPE Requirements:

- Safety Goggles
- Face Mask
- Do not wear jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Ensure mechanical fasteners are correctly tightened.
- 2) Wear required PPE (safety glasses, tie hair back, and put away jewelry)
- 3) One team member must complete this test procedure to ensure that the system is able to be carried by one person.

- 4) Zero/tare the weight scale if necessary.
- 5) Transfer entire feed drive system onto the weight scale.
- 6) Observe system weight. Verify that it is less than 50 lb.

Results: Pass Criteria, Fail Criteria, Number of samples to test

Pass Criteria: System weighs less than or equal to 50 lb.

Fail Criteria: System weighs over 50 lb.

Number of Samples: 1

CNC Feed Drive Test Procedure - Motor/Microcontroller

Test Name: Open-Loop Motor Output Verification.

Purpose: To verify table will move given an open-loop command.

Scope: This will test the communication between a desired command from GUI to the movement of the table.

Equipment:

- Entire powered feed drive system
- Surface for the system to be placed on
- Measuring Tool (Calipers)
- GUI Interface
- Microcontroller
- Power Supply
- Voltmeter
- Tape
- Sharpie
- Stopwatch

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <40lbs, 2 man lift <70lbs)
- Power Failures
- Electrical Shock

PPE Requirements:

- Safety Goggles
- Face Mask
- Stash away jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure:

- 1) Ensure electrical connections are secure and verify the system is grounded.
- 2) Ensure mechanical fasteners are correctly tightened.
- 3) Put on required PPE (safety glasses, tie hair back, and put away jewelry)
- 4) Record initial position (Put a piece of tape on the table and line it up with a Sharpie)
- 5) Turn on Power Supply
- 4) Give input voltage command to the GUI
- 5) Second team member starts stopwatch. Additional time measurement will be taken from GUI
- 6) Record final position
- 7) Record a time for when the table reaches desired distance
- 8) Repeat for increments from 0 V until max voltage
- 9) Repeat test 3 times to test repeatability

Results:

Pass Criteria: Pass if the result is within 5% percent of our desired speed.

Fail Criteria: Failed if the result is out of the 5% range or if the limit switches fail.

Number of samples to test: 1

Project Title: F12 CNC Feed Drive

1. Desired results with required uncertainty
 - a. We are looking for our motor encoder to have a resolution of + or – 1 PPR of the manufactures stated 28 PPR.
2. Diagram of apparatus and instrumentation
 - a. (On separate page)
 - b. Steps include placing toothpick vertically on motor spindle and apply voltage incrementally to cause the spindle to rotate a full revolution.
3. Priority list of measurements to be undertaken
 - a. We will need to measure 12.9° in increments to 360°
4. Schedule including calibration, zero/tare,baseline, repeats
 - a. At the end of the motor spindle will tape a toothpick vertically and set it to 0°.
 - b. We will repeat the measurement at 12° and 13°.
5. Data Analysis equations/spreadsheet with uncertainty

X_1	X_2	U_{x1}	U_{x2}	$C=f(x_m)$	$f(x_1+u_{x1})$	$f(x_2+u_{x2})$	S_1	S_2	u_c
# of rotations (rotations)	Angle (Θ)	U_{Voltage}	U_{Angle} Reading +/- 1	$C=x_2/x_1$	$f(x_1+.01)$	$f(x_1+1)$	S_{voltage}	S_{Angle}	U_c

6. Expected results

We expect to get right around 28 PPR and with this information we can validate that we will be getting the right distance for the students in the lab.

Test Procedure - Microcontroller Summary

Test Name: Microcontroller Test for all Outputs and Inputs

Purpose: *The purpose of this test is to ensure that we can accurately control the voltage outputs on the Nucleo Microcontroller and ensure that the Nucleo can accurately read from the input pins. Additionally, we must ensure serial communication between the computer and the microcontroller.*

Scope: The scope of this test is to ensure that the Nucleo Microcontroller is fully functional, able to accurately read from input pins and output predictable voltage. Verification is required of the USB connection, motor driver pin outputs, encoder pin outputs and inputs, and ADC readers for the limit switches.

Equipment:

- Jumper cables
- Multimeter
- Nucleo Microcontroller
- Motor Driver Breakout Board
- USB 2.0 cable

Hazards:

- Static discharge from hands to microcontroller
- Short circuit across board
- Electric shock to users

PPE Requirements:

- Avoid contact with electrical components with bare hands.
- DO NOT plug in any components without verifying wiring with teammates.

Facility: Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Connect the Motor Driver hardware to the Nucleo by stacking the breakout board onto the correct pins on the Nucleo.

2) Power on the Nucleo by connecting the computer to the Nucleo's local USB port with the USB 2.0 cable. Ensure that the jumper cable controlling Nucleo power on the board is connected to U5V, rather than E5V.

3) Ensure access to the Nucleo via PuTTY. Print a few lines to the PuTTY terminal to ensure board functionality.

Serial Communication

4) Push main.py from the "Test Folder" to the Nucleo storage by pushing via the Command Prompt.

5) Power off the Nucleo by unplugging the USB cord. Plug the Nucleo back in so that main.py runs on startup.

6) Test Serial Communication via UART to the Nucleo by running the UIFrontend.py file on the PC side. The Nucleo should return values via the Serial Port at the completion of UIFrontend.py.

Encoder Pins

7) Using a multimeter, ensure a 5V difference between the native board 5V pin and ground pin. This will be used to power the encoders and limit switches.

8) Set Pins B6 and B7 to read ADC input and short the 5V pin to each of the pins. Read from the ADC on each pin and verify a value of around 4095 (for a 12-bit ADC). These pins will be used to read data from the encoders.

Motor Pins

9) Set Pin A15 to be an output pin. Set the pin high and then low and use a multimeter to verify that the pin produces a voltage of 3.3V and 0V, respectively.

10) Initialize Pins B4 and B5 and initialize to do Pulse Width Modulation (PWM) from those pins. Verify that linearly varying duty cycle between 0 and 100 linearly varies the pin output voltage with a multimeter. Additionally, measure M- and M+ on the motor driver board and ensure that the voltage matches Pins B4 and B5 respectively.

Limit Switches

11) Set Pins A0 and A1 to read ADC input and short the 5V pin to each of the pins. Read from the ADC on each pin and verify a value of around 4095 (for a 12-bit ADC). These pins will be used to read from the proximity sensors.

Results: The test is passed if the Nucleo can successfully communicate with the computer via serial port, match pin outs with expected values, and read an external input via ADC on all relevant pins.

Test Procedure – User Interface

Test Name: User Interface (component of computer/microcontroller communication)

Purpose: *To verify that a new user can run the system in a reasonable amount of time.*

Scope: This test will verify that a new user is able to run our system through the user interface in five minutes or less. This will require an interface that is understandable and effective.

Equipment:

- Assembled feed drive system
 - o User Interface
- Stopwatch
- Roommate

Hazards:

- Rotating components
- "Heavy Lift" Items (1 man lift <70lbs)
- User's unfamiliarity with the system will present unforeseen hazards.

PPE Requirements:

- Safety Goggles
- Face Mask
- Do not wear jewelry
- Tie up loose hair
- 6 feet separation

Facility:

- Ryan and Caleb's House

Procedure: (List number steps of how to run the test, can include sketches and/or pictures):

- 1) Coordinate a time to complete this test procedure given roommate's availability.
- 2) Ensure mechanical fasteners are correctly tightened.

- 3) Wear required PPE (safety glasses, tie hair back, and put away jewelry)
- 4) Maintain close supervision of roommate throughout test procedure.
- 5) Present the feed drive system to roommate. Inform them of all associated hazards.
- 6) Take note of any unforeseen hazards/sticking points of the roommate throughout the test procedure. Record them in the **Test Notes** section below.
- 7) Without directly instructing roommate, start time on the stopwatch and have the roommate attempt to produce motion of the system through the user interface.
- 8) The roommate must be able to produce motion of the table in under five minutes.

Test Notes:

-

Results: Pass Criteria, Fail Criteria, Number of samples to test

Run test procedure with the maximum plate loading of 5kg.

Pass Criteria: Roommate safely runs the table in under five minutes by using the user interface without direct instruction.

CNC Feed Drive

Owner's Manual

Written by the M.E Senior Project Team

Nicholas DeSimone, Ryan Funchess, Juan Majano, and Caleb O’Gorman

Required Personal Protective Equipment

- Safety Glasses

Operation Safety Rules

- Tie up loose hair.
- Remove loose clothing.
- Remove loose jewelry and set it aside.

Before Powering the System

Follow these instructions to ensure the necessary safety measures are in place and components are in their correct positions before the system can receive power.

Placing the Feed Drive Assembly

The feed drive assembly, including all components, contained within the aluminum frame and as pictured below, must be placed appropriately atop the wooden base plate before it can be connected to power. If the feed drive assembly is not in place atop the wooden base, carry out the following steps.



Figure 1: Isometric view of the feed drive assembly, including mechanical and electrical components.

Caution: Do **NOT** lift the assembly by the motor or any other exposed components. Only lift the assembly from beneath the base plate with full support on either end.

1. Ensure that the wooden base is located on the surface where it will be operated, which should be flat, clear of debris, and immobile.
2. Two users pick up and support each end of the assembly from underneath the base plate. Note any pinch points and avoid them accordingly, although none should be present when the users support the assembly from underneath the base plate (as spacing is provided by the legs of the assembly).
3. Place the feed drive atop the wooden base such that each of its four corners is aligned in the four designated marks on the wooden base.

Checking Electrical Connections

Before the use of the system, verify that all electrical connections are rigid on the Nucleo and the electro-mechanical components. The eight wires connected to the Nucleo/Motor Driver are tabulated below, with pictures corresponding to the correct placement of these components.

Table 1: Wiring connections from Nucleo/Motor Driver to all other components.

Pin on the Nucleo/Motor Driver	Wire Destination
Pin A6	Limit Switch Output (Motor Side)
GND	Limit Switch Input (Ball Bearing Side)
Motor Driver Channel A +	Motor +
Motor Driver Channel A-	Motor -
3.3V Pin	Vcc on Encoder
GND	GND on Encoder
Pin A1	Encoder Channel A
Pin A0	Encoder Channel B

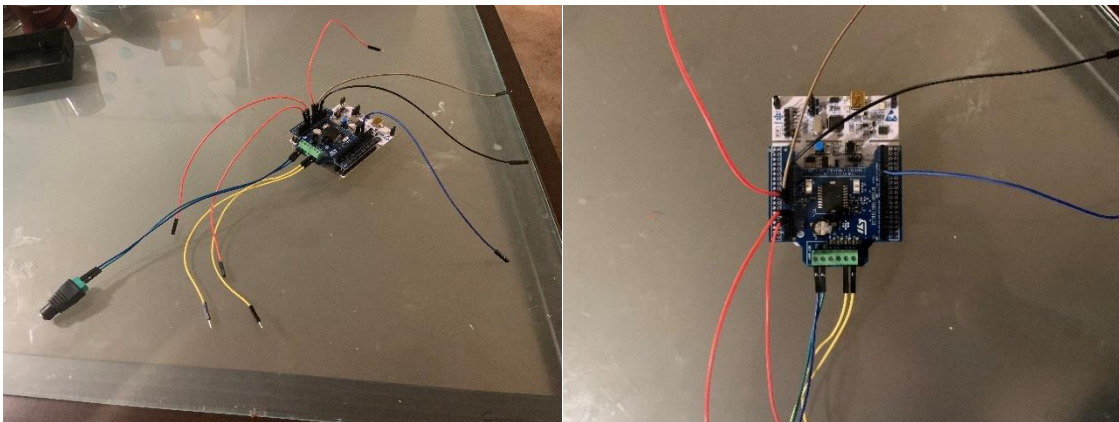


Figure 2: Proper wire electrical connections on the Nucleo Side

For each of the above connections, ensure that the cables are rigidly connected at both ends and the cable is not frayed or torn.

Attaching the Lid

The clear plastic storage bin must be placed atop and engaged with the four switches of the wooden base plate before the system can receive power.

1. Align the clear plastic “lid” over the feed drive on the wooden base plate.
2. Place lid over system, taking care to not unplug any wires attached to the motor or limit switches.

Final Checks

Finally, a few final mechanical components must be checked:

1. Check that all screws are fully fastened.
2. Ensure that there are no obstructions in the path of the ballscrew.
3. Check that the ballscrew appears to be lubricated. System performance is strongly directed by lubrication on the ballscrew. If the ballscrew does not appear damp, squirt machine oil over the top of the ballscrew until the screw drips slightly onto the base plate. Mesh this with the system by powering down the feed drive and turning the shaft coupling such that the table moves the length of the screw.
4. Check to be sure that both shaft coupling hubs are fully engaged, as seen in Figure 3 below. If hubs are not full engaged, lightly tap the ball bearing until hubs are engaged.

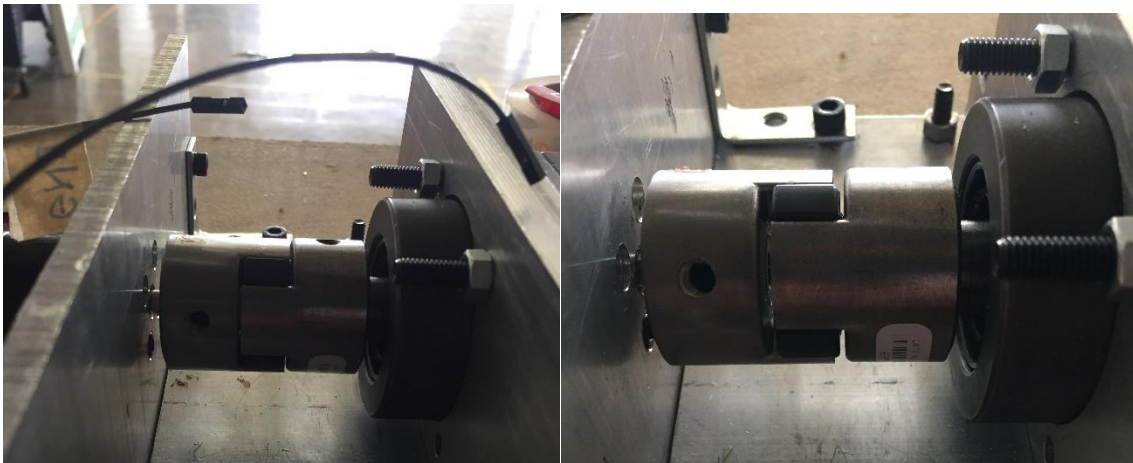


Figure 3: Side by side images depicting fully engaged coupling hubs (left) and non-fully engaged coupling hubs (right).

Setting up the Feed Drive for Use

After the system has been inspected, the code has been set up, and all the initial safety steps have been completed the system can finally be used.

1. Ensure that the lid is closed.
2. Ensure that the motor driver input power is plugged into the wall adapter.



Figure 4: Motor Driver input power connected to the DC power supply to be plugged into the wall.

3. Plug in the wall adapter to a standard 100-240V wall outlet
4. Plug the Nucleo into the computer via the USB. This cord is used to power both the Nucleo as well as allow for serial communication between the MATLAB frontend and the Nucleo Backend. Upon startup, the Nucleo will zero the system to the limit switches.

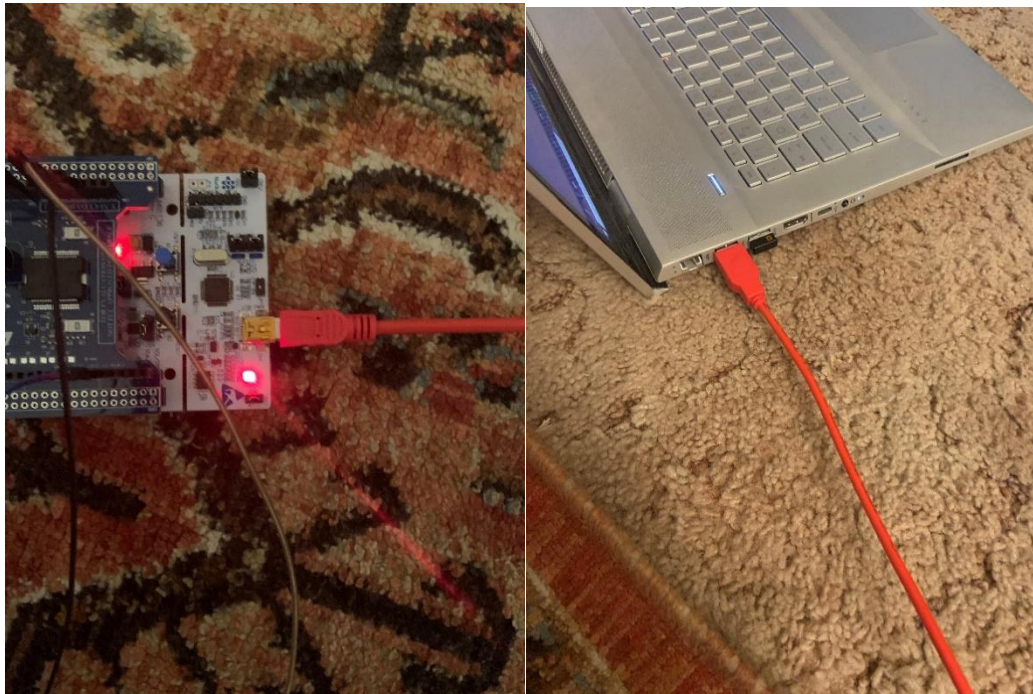


Figure 5: USB Cable connecting Nucleo to Computer.

5. Return to the computer and initiate the process.

Caution: To **STOP** the process at any time press the reset button on the controller or unplug the motor driver from the controller enclosure.

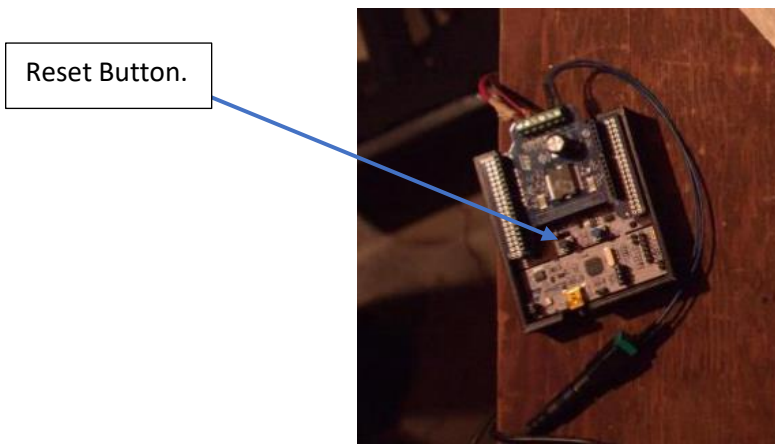


Figure 6: Shows the location of the reset button on Nucleo.

Operating the User Interface-UPDATE

Caution: If you hear a high-pitched squealing noise from the motor at any point during operation, there is excessively high current running through the motor and it has stalled. If this occurs, press the black “RESET” button on the Nucleo and start over on the user interface. If this occurs multiple times, check to see if the ballscrew requires extra lubrication and check to ensure your gains/open loop duty cycle are within the specified range.

The user interface for the CNC feed drive is written in MATLAB and thus will be controlled from a computer attached to the feed drive. The following steps cover how to control your feed drive with the MATLAB user interface:

1. Before attempting to run the system, ensure that the COM Port in the MATLAB Source Code matches the COM Port relating to the Nucleo. The COM Port relevant to the Nucleo can be found in the “Windows Device Manager”, as shown in Figure 7 below.

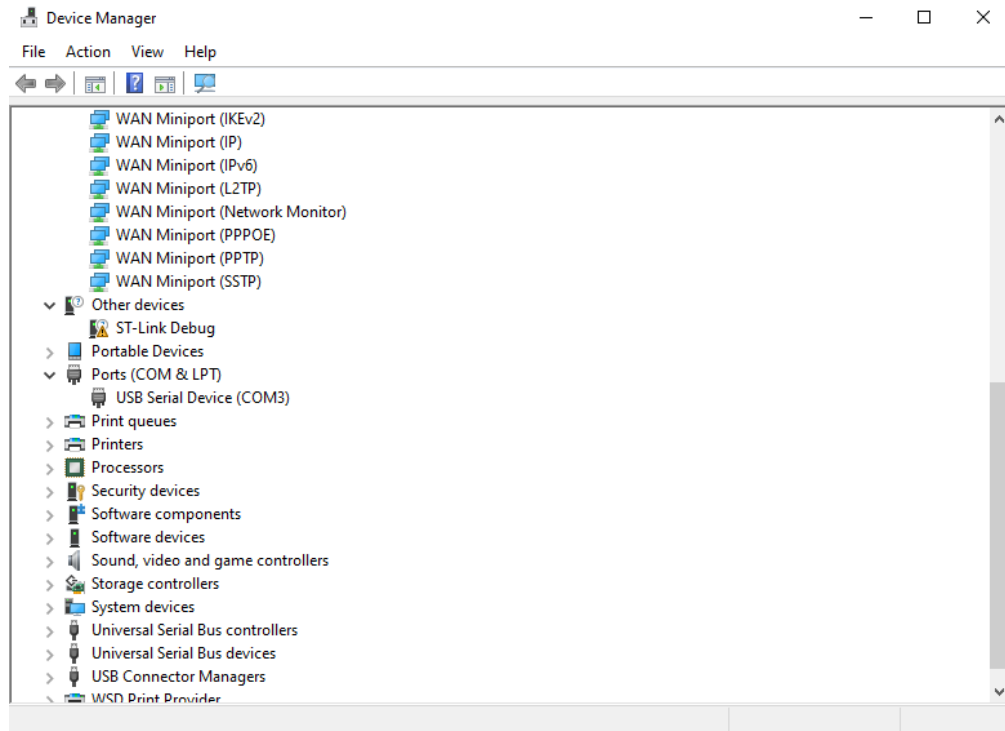


Figure 7: Windows Device Manager showing that the Nucleo is connected to COM3. This value will be changed as commented in the MATLAB Source Code.

2. Launch the MATLAB App titled “CNC Feed Drive GUI” from the MATLAB apps taskbar, as seen in Figure 8 below.

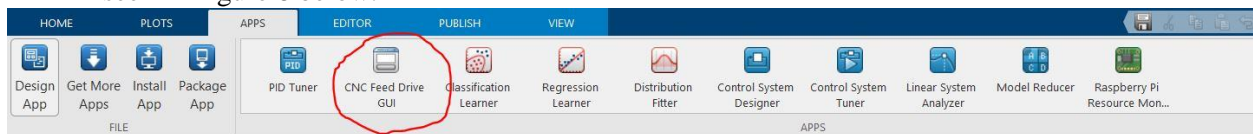


Figure 8: Location of “CNC Feed Drive GUI” in the MATLAB apps taskbar, which appears once the installation file is run.

- When the app launches, the GUI will be presented as shown in Figure 9 below.



Figure 9: Graphical User Interface (GUI) run through the “CNC Feed Drive GUI” App

- From here, there are two options for how the system will be run:
 - Open Loop Control:** This involves the table zeroing out at a limit switch and then moving at the requested duty cycle until a travel distance of 20 cm is reached. To use this mode, simply leave the closed loop box unchecked and enter in a duty cycle. Since we are running a 12V motor from a 24V source, the maximum applied duty cycle to the motor is 50%, which is regulated in software on the backend.
 - Closed Loop Control:** This mode involves PID control of the system. The table zeroes out to the limit switch and then approaches a series of three requested positions. Once steady state is reached at one positions, the next position is set as the reference. To run in PID closed loop with the GUI, input a value for each of the three controller gains on the left side of the GUI, as well as three linear positions between 0.5 and 25 cm, separated by commas.

Note: Overly large controller gains could cause unprecedented system response and thus it is imperative to monitor the feed drive during operation to halt it if necessary. We recommend keeping all gains between 0 and 10.

- Press “Enter” to begin real time data collection.
- The data for the table's motion will be displayed in the axes on the bottom of the application. A time series of the data will also appear in the MATLAB workspace. To save this data, right-click on the workspace and select “Save.”
- After you are done and have ensured your data is saved, closeout of the GUI.

Attention: If the table contacts the limit switch and stops, this is part of normal operation, and it signifies that the system overshoot is too high to be acceptable. When a limit switch is contacted, the motor is disabled. If a switch is contacted during your experiment, unplug the Nucleo and Motor Driver and carefully turn the shaft coupling by hand until the table is an acceptable position for the experiment to be restarted.

Powering Down System

Once the program has gone through its cycle and data has been collect follow the steps below:

1. Unplug the system from the wall outlet.
2. Unplug the Nucleo from the computer
3. Close lid.

Parts List

Table 2: List of parts with their respective vendors

Items Purchased	Vendor
PLA Filament	Amazon
Ball Screw, Ballnut, and Bearings	Automation4Less
Rails and linear bearings assembly	Vevor
NUCLEO-L476RG (Microcontroller)	ST Microelectronics
Limit Switches and shaft couplings	McMaster-Carr
Aluminum Sheets	Grainger
Motor Driver	ST Microelectronics
Power Supply	Amazon
Roller Switches	Amazon
Motor Driver	ST
56 Fasteners and Corner Brackets <ul style="list-style-type: none"> • M6x0.8mm Socket Head screws and washers for ballnut to table assembly • M6 Corner Bracket. 	Ace Hardware
Female Bullet Adaptors	ServoCity
Electric Motor	ServoCity

Maintenance

- Apply machine oil at least once a week.
 - Apply small amount spread out along ballscrew such that until the screw drips slightly onto the base plate. Mesh this with the system by powering down the feed drive and turning the shaft coupling such that the table moves the length of the screw.
- Inspect bearings for wear. If so:
 - Contact Lab Technician for replacement.
 - Or review manufacturing plan for assembly.
- If limit switches are faulty:
 - Obtain replacement parts from parts list follow manufacturing plan to re-install.
- Check fasteners once a quarter for wear. Replace if necessary.

Troubleshooting Guide

For any issues related to the use and or operation of this system please contact Dr. Siyuan (Simon) Xing:

sixing@calpoly.edu

A.21 MATLAB GUI Code

```
classdef app1 < matlab.apps.AppBase

% Properties that correspond to app components
properties (Access = public)
    UIFigure matlab.ui.Figure
    UserControlPanel matlab.ui.container.Panel
    kp matlab.ui.control.NumericEditField
    ki matlab.ui.control.NumericEditField
    kd matlab.ui.control.NumericEditField
    ControllerLabel matlab.ui.control.Label
    KpLabel matlab.ui.control.Label
    KiLabel matlab.ui.control.Label
    KdLabel matlab.ui.control.Label
    StepInput matlab.ui.control.EditField
    StepInputControlLabel matlab.ui.control.Label
    cmLabel matlab.ui.control.Label
    EnterButton matlab.ui.control.Button
    ClosedLoopCheckBox matlab.ui.control.CheckBox
    LeaveuncheckedforOpenLoopLabel matlab.ui.control.Label
    DutyCycle matlab.ui.control.NumericEditField
    DutyCycleLabel matlab.ui.control.Label
    DataDisplayPanel matlab.ui.container.Panel
    DataPlot matlab.ui.control.UIAxes
    SpeedPlot matlab.ui.control.UIAxes
end

properties (Access = public)
    ser = serialport("COM7", 115200);
```

```

end
methods (Access = private)
function data_collection(app)
while app.ser.NumBytesAvailable == 0
% Wait for length
end
times = zeros(1, 2000);
pos = zeros(1, 2000);
speed = zeros(1, 2000);
rd = "";
n = 1;
while rd ~= "DION"
rd = readline(app.ser);
% cd = readline(app.ser);
% dd = readline(app.ser);
% ed = readline(app.ser);
% message = rd + cd + dd + ed;
if rd == "DION"
continue
end
% app.Label.Text = message;
% app.Label2.Text = string(app.ClosedLoopCheckBox.Value);
ss = split(rd, ',');
times(n) = str2double(ss(1))/1000;
pos(n) = str2double(ss(2));
speed(n) = str2double(ss(3));
plot(app.DataPlot,times(1:n),pos(1:n))
plot(app.SpeedPlot,times(1:n),speed(1:n))
n = n+1;
end
assignin('base', 'speed', speed)

```

```

assignin('base', 'positions', pos)
assignin('base', 'time', times)
%app.TimeLabel.Text = num2str(times);
%app.VallLabel.Text = num2str(vals);
end
end

% Callbacks that handle component events
methods (Access = private)

% Code that executes after component creation
function starter(app)
flush(app.ser)
configureTerminator(app.ser, "CR/LF")
end

% Button pushed function: EnterButton
function EnterButtonPushed(app, event)
if app.StepInput.Value == "splash"
app.splash_mode();
else
s = ",";
kps = string(app.kp.Value);
kis = string(app.ki.Value);
kds = string(app.kd.Value);
sts = string(app.StepInput.Value);
if app.ClosedLoopCheckBox.Value == true
ols = "1";
else
ols = "0";

```

```

end
dts = string(app.DutyCycle.Value);
val = kps + s + kis + s + kds + s + sts + s + ols + s + dts;
write(app.ser, val, "string");
app.data_collection();
end
end

% Close request function: UIFigure
function closer(app, event)
delete(app.ser)
delete(app)
end
end

% Component initialization
methods (Access = private)

% Create UIFigure and components
function createComponents(app)

% Create UIFigure and hide until all components are created
app.UIFigure = uifigure('Visible', 'off');
app.UIFigure.Position = [100 100 650 516];
app.UIFigure.Name = 'MATLAB App';
app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @closer, true);

% Create UserControlPanel
app.UserControlPanel = uipanel(app.UIFigure);
app.UserControlPanel.TitlePosition = 'righttop';

```

```
app.UserControlPanel.Title = 'User Control';  
app.UserControlPanel.BackgroundColor = [1 0.8902 0.5843];  
app.UserControlPanel.Position = [1 312 650 205];
```

```
% Create kp
```

```
app.kp = uieditfield(app.UserControlPanel, 'numeric');  
app.kp.HorizontalAlignment = 'center';  
app.kp.Position = [117 128 51 28];  
app.kp.Value = 6;
```

```
% Create ki
```

```
app.ki = uieditfield(app.UserControlPanel, 'numeric');  
app.ki.HorizontalAlignment = 'center';  
app.ki.Position = [118 80 51 28];  
app.ki.Value = 2;
```

```
% Create kd
```

```
app.kd = uieditfield(app.UserControlPanel, 'numeric');  
app.kd.HorizontalAlignment = 'center';  
app.kd.Position = [117 32 51 28];  
app.kd.Value = 1;
```

```
% Create ControllerLabel
```

```
app.ControllerLabel = uilabel(app.UserControlPanel);  
app.ControllerLabel.HorizontalAlignment = 'center';  
app.ControllerLabel.FontSize = 16;  
app.ControllerLabel.Position = [105 155 75 22];  
app.ControllerLabel.Text = 'Controller';
```

```
% Create KpLabel
```

```

app.KpLabel = uilabel(app.UserControlPanel);
app.KpLabel.HorizontalAlignment = 'center';
app.KpLabel.Position = [131 107 25 22];
app.KpLabel.Text = 'Kp';

% Create KiLabel
app.KiLabel = uilabel(app.UserControlPanel);
app.KiLabel.HorizontalAlignment = 'center';
app.KiLabel.Position = [130 59 25 22];
app.KiLabel.Text = 'Ki';

% Create KdLabel
app.KdLabel = uilabel(app.UserControlPanel);
app.KdLabel.HorizontalAlignment = 'center';
app.KdLabel.Position = [130 11 25 22];
app.KdLabel.Text = 'Kd';

% Create StepInput
app.StepInput = uieditfield(app.UserControlPanel, 'text');
app.StepInput.Position = [427 114 65 22];
app.StepInput.Value = '10,15,5';

% Create StepInputControlLabel
app.StepInputControlLabel = uilabel(app.UserControlPanel);
app.StepInputControlLabel.HorizontalAlignment = 'center';
app.StepInputControlLabel.FontSize = 14;
app.StepInputControlLabel.Position = [359 155 119 22];
app.StepInputControlLabel.Text = 'Step Input Control';

% Create cmLabel

```



```

app.cmLabel = uilabel(app.UserControlPanel);
app.cmLabel.Position = [526 114 29 22];
app.cmLabel.Text = '(cm)';

% Create EnterButton
app.EnterButton = uibutton(app.UserControlPanel, 'push');
app.EnterButton.ButtonPushedFcn = createCallbackFcn(app, @EnterButtonPushed,
true);
app.EnterButton.Position = [298 114 100 22];
app.EnterButton.Text = 'Enter';

% Create ClosedLoopCheckBox
app.ClosedLoopCheckBox = uicheckbox(app.UserControlPanel);
app.ClosedLoopCheckBox.Text = 'Closed-Loop';
app.ClosedLoopCheckBox.Position = [298 49 90 22];
app.ClosedLoopCheckBox.Value = true;

% Create LeaveuncheckedforOpenLoopLabel
app.LeaveuncheckedforOpenLoopLabel = uilabel(app.UserControlPanel);
app.LeaveuncheckedforOpenLoopLabel.Position = [298 19 185 22];
app.LeaveuncheckedforOpenLoopLabel.Text = '*Leave unchecked for Open-
Loop';

% Create DutyCycle
app.DutyCycle = uieditfield(app.UserControlPanel, 'numeric');
app.DutyCycle.Position = [427 49 65 22];

% Create DutyCycleLabel
app.DutyCycleLabel = uilabel(app.UserControlPanel);
app.DutyCycleLabel.Position = [526 49 86 22];
app.DutyCycleLabel.Text = 'Duty Cycle (%)';

```

```

% Create DataDisplayPanel
app.DataDisplayPanel = uipanel(app.UIFigure);
app.DataDisplayPanel.TitlePosition = 'righttop';
app.DataDisplayPanel.Title = 'Data Display';
app.DataDisplayPanel.BackgroundColor = [1 0.8902 0.5843];
app.DataDisplayPanel.Position = [1 1 650 312];

% Create DataPlot
app.DataPlot = uiaxes(app.DataDisplayPanel);
title(app.DataPlot, 'Pos vs Time')
xlabel(app.DataPlot, 'Time (s)')
ylabel(app.DataPlot, 'Pos (cm)')
zlabel(app.DataPlot, 'Z')
app.DataPlot.PlotBoxAspectRatio = [7.95652173913043 1 1];
app.DataPlot.Position = [1 153 633 128];

% Create SpeedPlot
app.SpeedPlot = uiaxes(app.DataDisplayPanel);
title(app.SpeedPlot, 'Speed vs Time')
xlabel(app.SpeedPlot, 'Time (s)')
ylabel(app.SpeedPlot, 'Speed (cm/s)')
zlabel(app.SpeedPlot, 'Z')
app.SpeedPlot.PlotBoxAspectRatio = [7.79787234042553 1 1];
app.SpeedPlot.Position = [0 24 634 130];

% Show the figure after all components are created
app.UIFigure.Visible = 'on';

end

end

```

```

% App creation and deletion
methods (Access = public)

    % Construct app
    function app = app1

        % Create UIFigure and components
        createComponents(app)

        % Register the app with App Designer
        registerApp(app, app.UIFigure)

        % Execute the startup function
        runStartupFcn(app, @starter)

        if nargin == 0
            clear app
        end
    end
end

% Code that executes before app deletion
function delete(app)

    % Delete UIFigure when app is deleted
    delete(app.UIFigure)
end
end
end
end

```