Aleksi Hieta
Leif Miller
Ryan Bohl

Developer Guide

Abstract:

The Tribble is contained within a triangular structure that has been manufactured out of wood. On each edge of the triangular base there is a motor that rotates at an independent speed and direction to wind or dispense wires with its attached pulley. These three wires from these motors attach to each point of the central, triangular payload which holds the drawing utensil. There is also a depth sensor on the payload which measures the distance that the payload is away from the paper. Through controlling the three motors, the payload has accurate triple-axis movement. The user can input commands into a MATLAB interface to select the various shapes, which become displayed on a paper in the interface before they are drawn on the physical paper. The GUI communicates with a custom PCB which houses an Arduino Nano. The Arduino sends signals to three motors drivers to control them according to the user input.

Electrical Specifications:

- **Mini USB to Arduino: 5 volt 0.5 Amps**

This is for the serial connection to the arduino. Most computer USB ports operate within this specification

- **Barrel Jack: 12v, 3.5A (from Netgear AC adaptor)**

Our code has been written so that the accuracy of the motors is maximized at 12 volts. THe L293 drivers accept up to 36 volts, but the code would have to be changed if one were to supply greater power to the drivers.

User Guide:

To build the Tribble some basic woodworking tools are required:

- Power Drill
- Circular Saw
- Jig Saw
- Drill Bit Index Imperial
- Tape Measure Imperial
- Rafter Square Imperial
- C Clamp Set
- Sander

The first step was to cut all wood pieces to size. This involved measuring 60 degree angles for the enclosure and the payload. Clamping two halves of the particle board together before performing the cuts for the base ensures that both triangles are the same size. This goes for the payload as well. The top layer platform is then cut to accommodate the piece of paper. Each pillar is cut to size as well.

The enclosure is then prepped for assembly by drilling pilot holes for all screw points. This helps to ensure that the wood does not crack when screwed together. After assembling the enclosure eye hooks, pipe clamps, and the PCB can be attached.

After the payload is cut out, clamp the triangles together for hole drilling. Once drilled and assembled the ultrasonic sensor module is attached with nails.

The final setup step is to run all wires to the motor control PCB and attach the payload to the motors with the fishing line.

Page 20 onwards contains all of the information regarding PCB assembly and connections. Figure 22 shows a completed PCB with all of the necessary components and connections. View Figure 19 to ensure that all pin connections are correct to the arduino.

To begin using the tribble, download the 3 MATLAB code files on the showcase website and open them all in the MATLAB IDE. This is a plug and play solution that allows the user to quickly begin drawing with the Tribble

Artifacts (one paragraph desc. each)

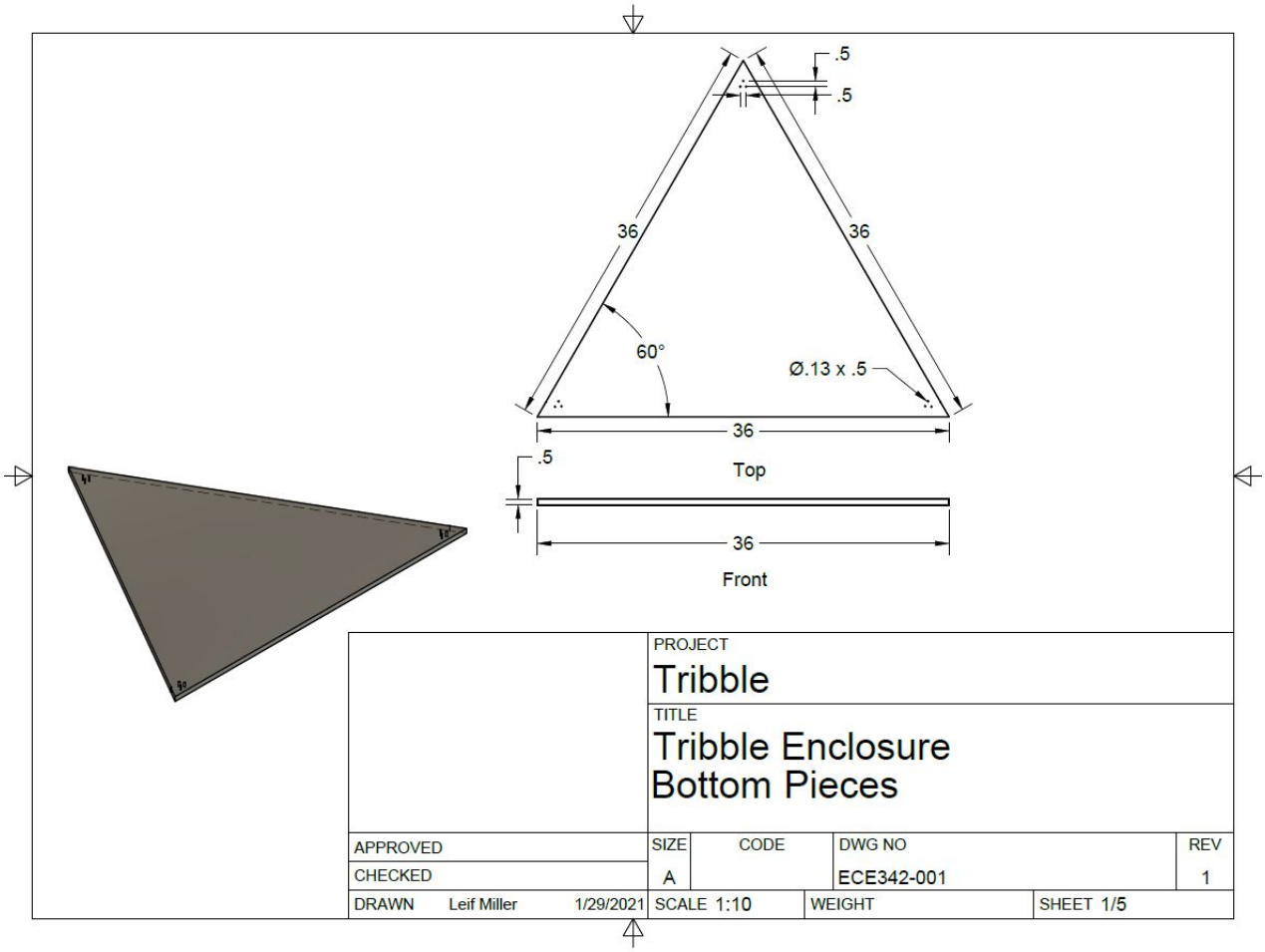

Figure 1: Bottom layer of enclosure

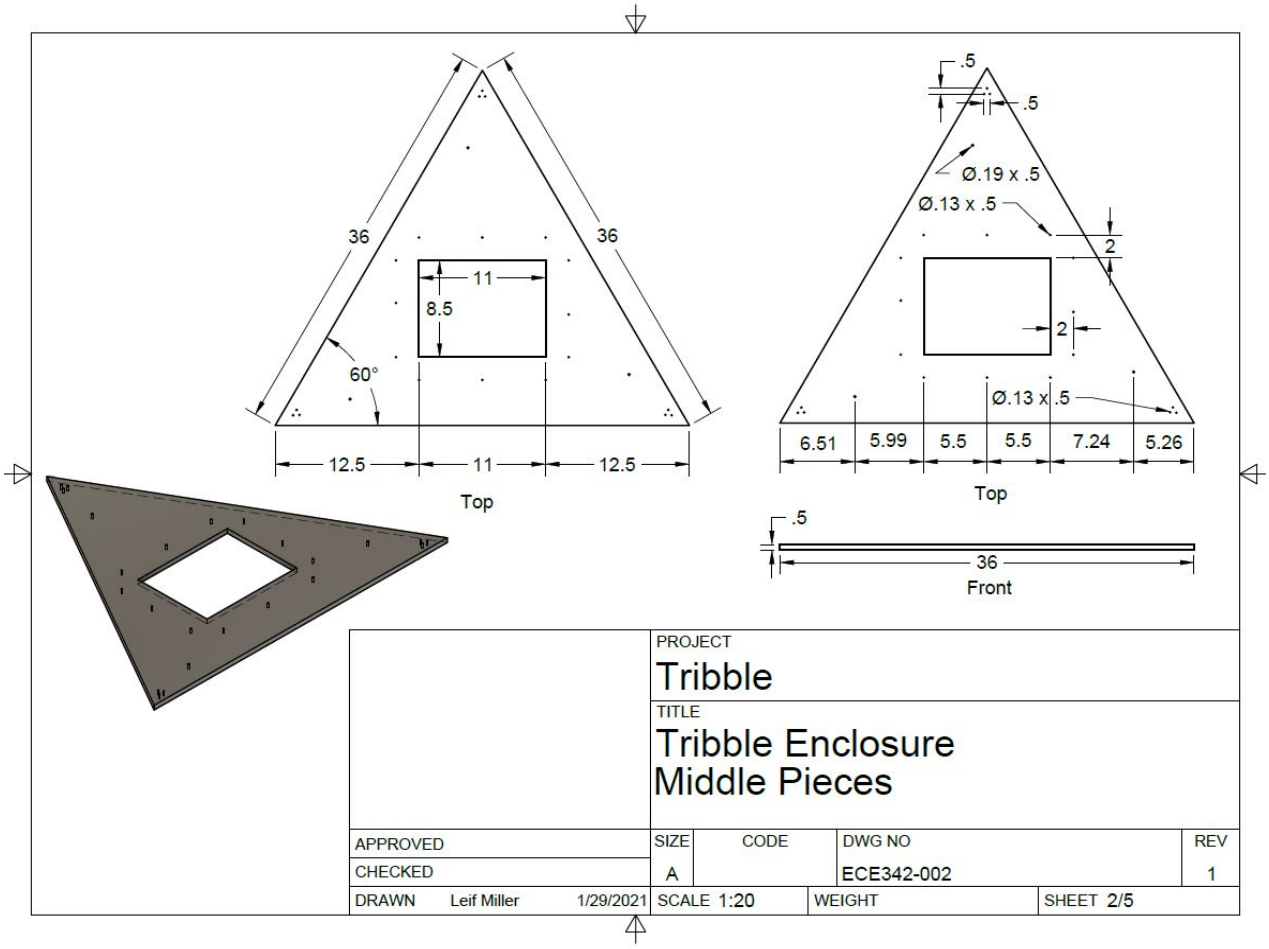

Figure 2: Top layer of enclosure

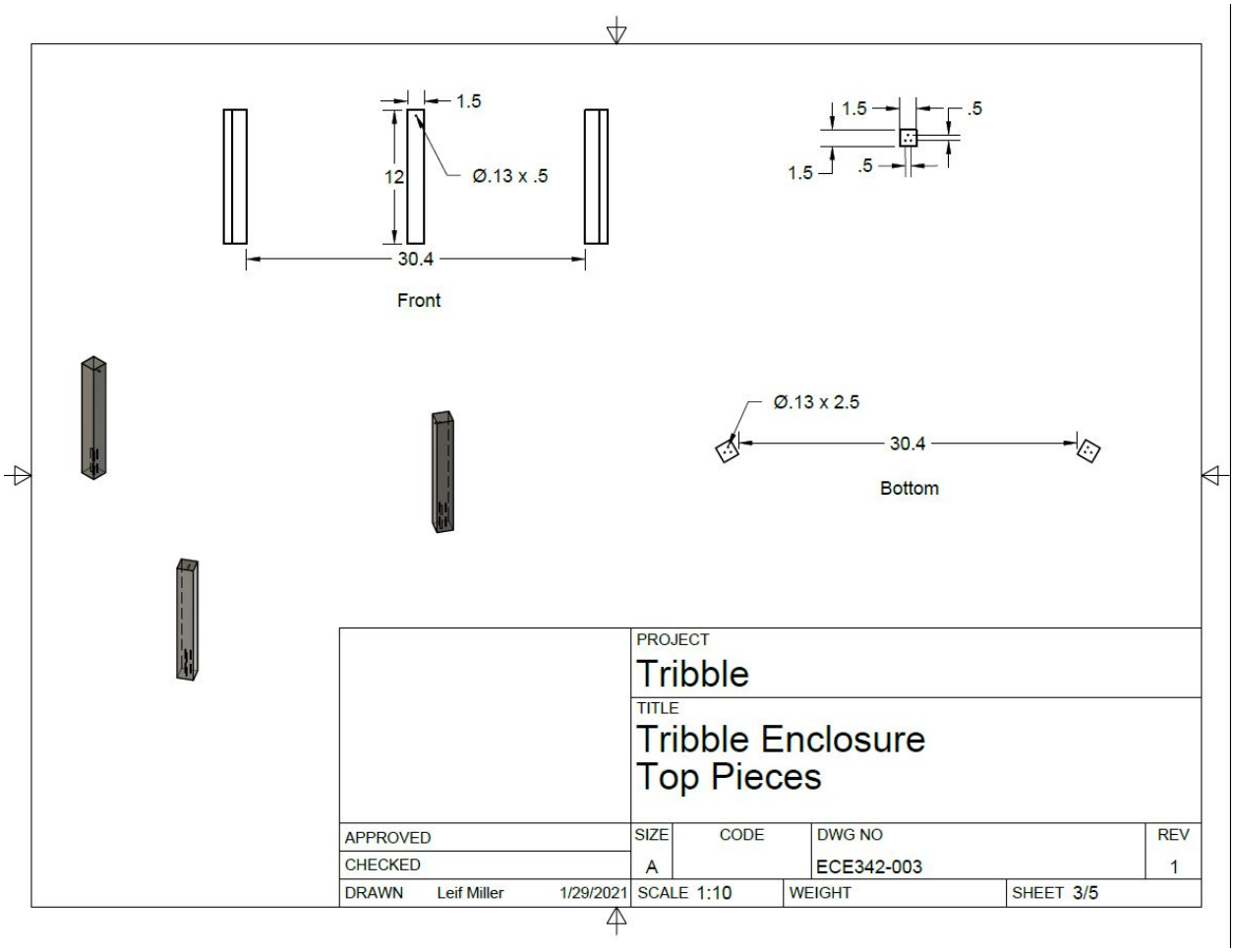

Figure 3: Pylons of enclosure

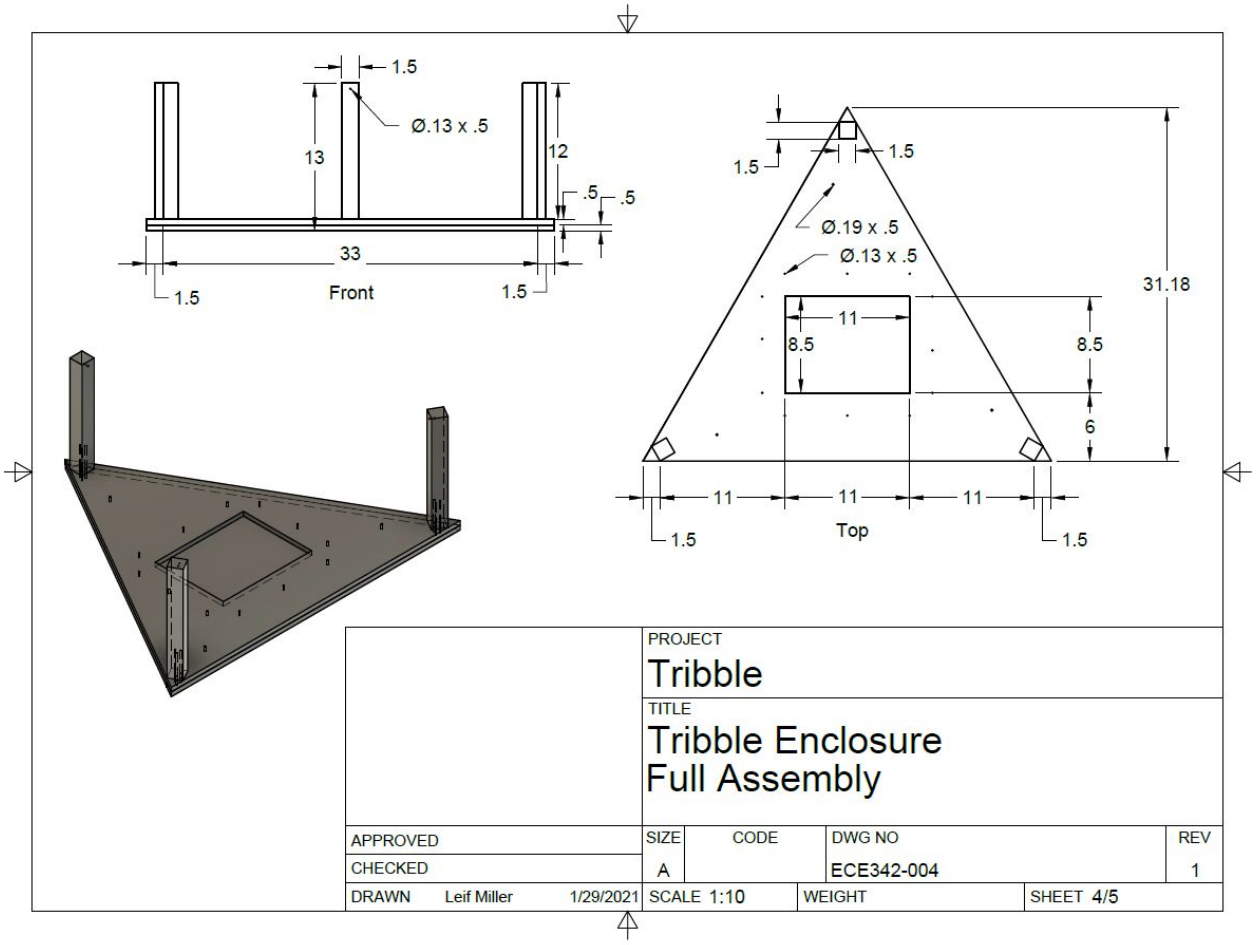

Figure 4: Complete enclosure CAD diagram

Tribble Enclosure Drawings:

These drawings contain information about the main Tribble enclosure. The main enclosure consists of a base platform, a platform with paper cutout, and three pillars. The two platforms are constructed of 1/2-inch particle board and the pillars are 2-inch by 2-inch boards. The enclosure is an equilateral triangle with provisions for standard paper retention, motor placement, and line elevation. The motor control PCB is attached above the paper cutout when viewed from the top.

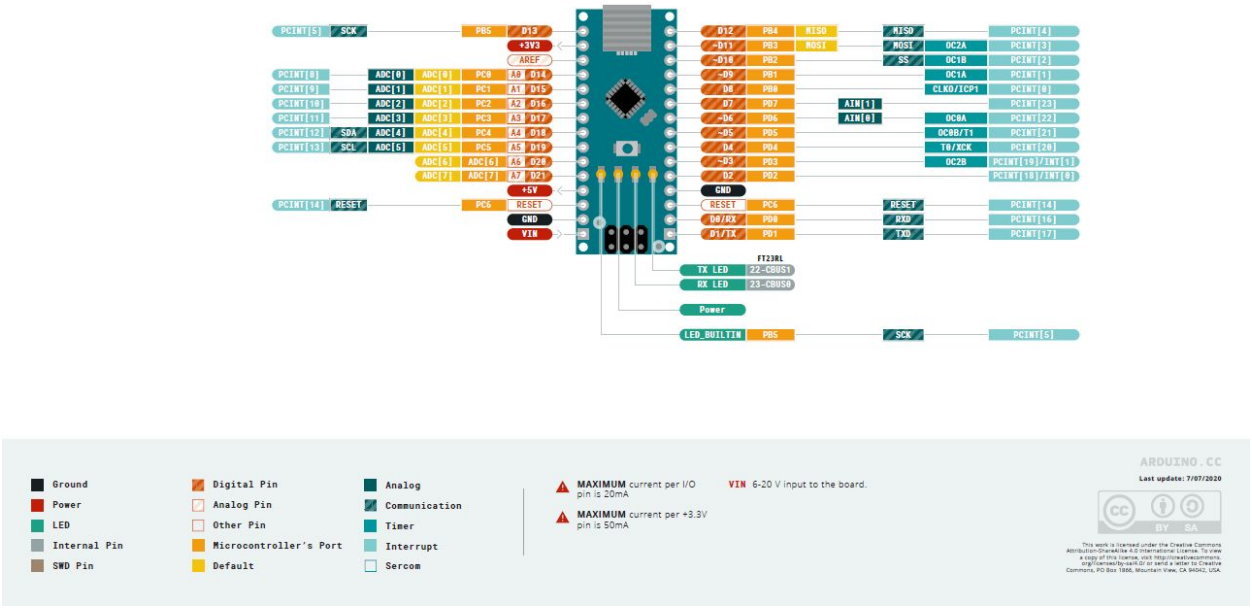

Figure 5: Arduino Nano Pinout

Figure 5 is a graphic showing the Arduino Nano Pinouts. The Arduino Nano utilizes the ATmega328P microprocessor and is used in this project for communication between the user interface and motor control.

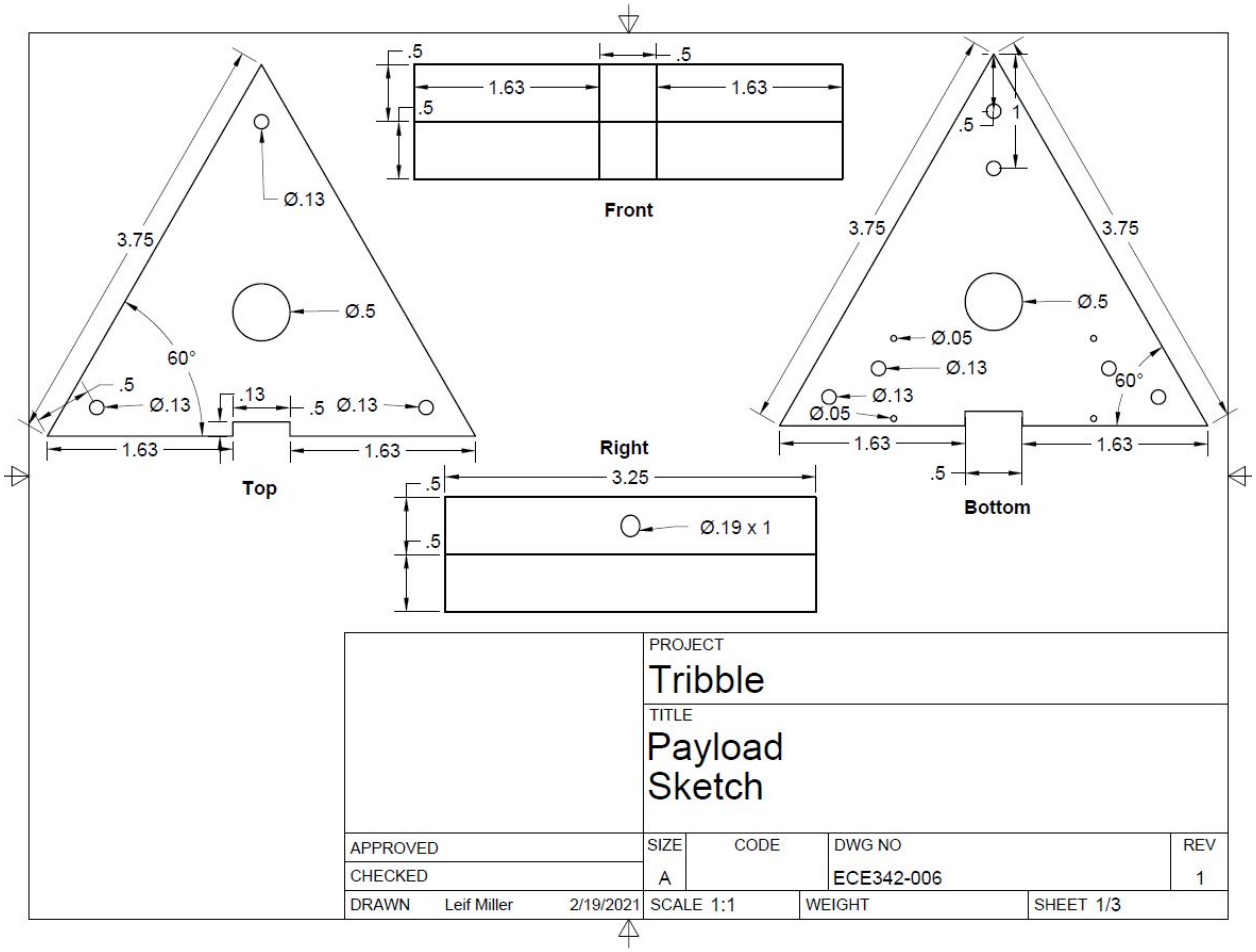

Figure 6: Drilled holes in Payload

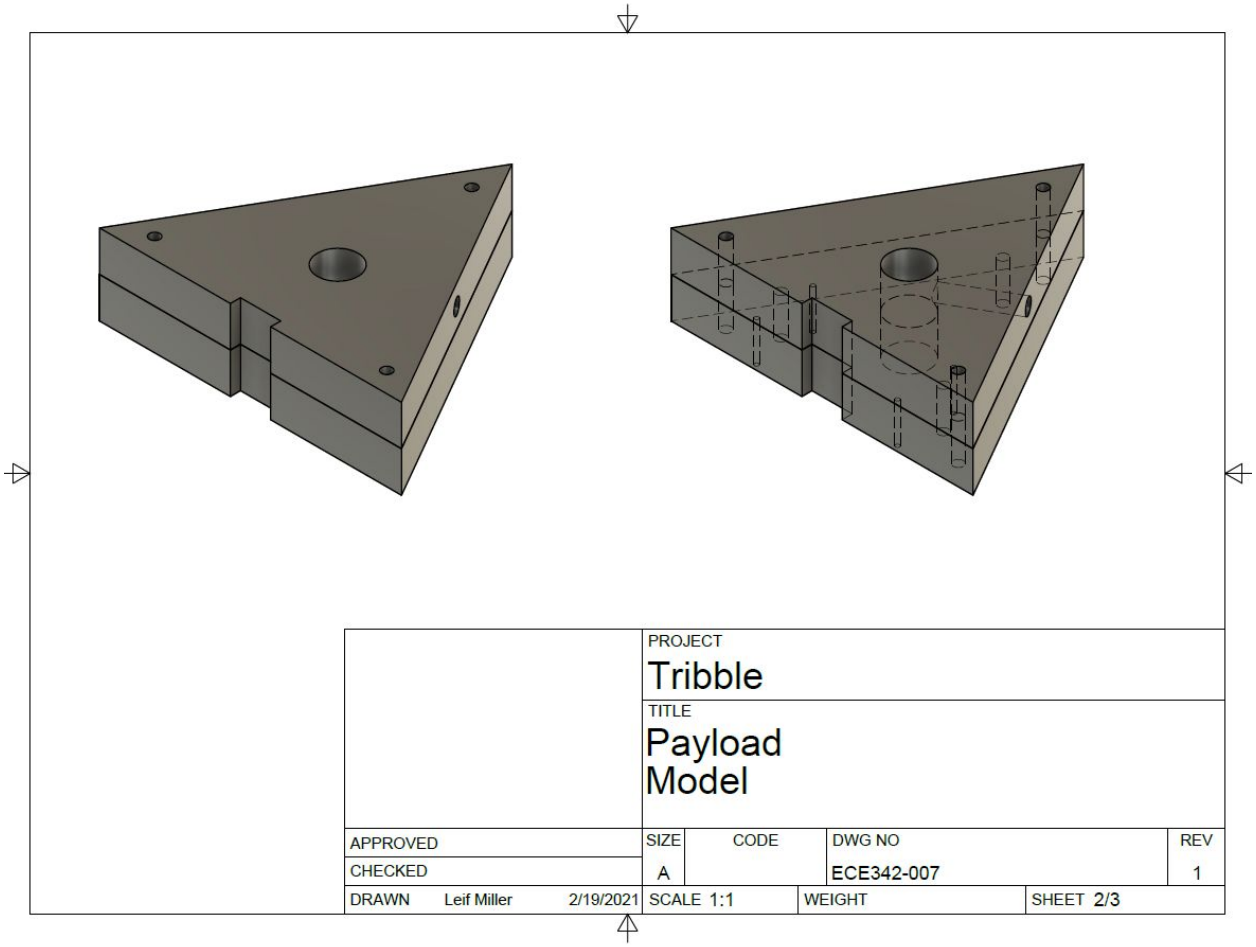

Figure 7: Complete Payload CAD Diagram

Tribble Payload Drawings:

The payload consists of two layers and an ultrasonic distance sensor. Both layers are constructed from 1/2-inch particle boards. Holes are drilled at each corner of the triangle to allow the fishing line to attach to the payload. The center hole is for the writing implement that is then retained with a horizontal set screw. The small cutout in the front of the payload is extra space for the pins of the ultrasonic distance sensor.

Part Information:

Part Name	Part Number	Qty	Ordered From	Cost Per Unit	Cost Total
1.6in Metal Eye Hook	AS-2001092126	7	https://www.amazon.com/gp/product/B083P4GQ7Q/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&th=1	$0.20	$1.40
Braided Fishing Line 30lb	B07TVSCLZC	1 roll	https://www.amazon.com/gp/product/B07TWW4CZV/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&th=1&psc=1 (Price per foot)	$0.03	$0.45
1in Pipe Clamp Rubber Cusioned	LK-88	3	https://www.amazon.com/gp/product/B01HPE185E/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&psc=1	$0.75	$2.25
GT2 Timing Pulley 4mm Bore	735314136076	3	https://www.amazon.com/gp/product/B07K7B8XHN/ref=ppx_yo_dt_b_asin_title_o04_s01?ie=UTF8&psc=1	$1.00	$3.00
Particle Board 1/2in x 4ft x 8ft	12262	1	Lowes in store purchase	$19.36	$19.36
2in x 2in x 6ft Douglas Fir	25533	1	Lowes in store purchase	$7.97	$7.97
2 1/2in Decking Screw	1147887	9	Lowes in store purchase	$0.09	$0.81
3/4in Brass Countersunk Screw	9035052	15	Already on hand	$0.13	$1.95
1/2in Hex Washer Head Screw	S35E#10050-100	3	Already on hand	$0.12	$0.36
Bolt 1 1/4-inch	43092Q	1	Ace Hardware in store purchase	$0.99	$0.99
Nail 3/4-inch	122622	4	Ace Hardware in store purchase	$0.04	$0.16
Ultrasonic Module	HC-SR04	1	https://www.amazon.com/gp/product/B01JG09DCK/ref=ppx_yo_dt_b_asin_title_o04_s02?ie=UTF8&psc=1	$1.92	$1.92
Motor Driver IC	L293D	3	https://www.amazon.com/gp/product/B00ODQM8KC/ref=ppx_yo_dt_b_asin_title_o04_s01?ie=UTF8&psc=1	$0.90	$2.70
5.5x2.1mm Female DC Power Jack	a15012900ux0190	1	https://www.amazon.com/gp/product/B011HFLKI2/ref=ppx_yo_dt_b_asin_title_o04_s01?ie=UTF8&psc=1	$0.58	$1.16
Shottky Diode 60V 2A	SB260	12	https://www.amazon.com/gp/product/B082H86136/ref=ppx_yo_dt_b_asin_title_o04_s01?ie=UTF8&psc=1	$0.28	$3.36
Encoder Motor 12V DC 300 RPM	TS-25GA370	3	https://www.amazon.com/gp/product/B07GNGCNK1/ref=ppx_yo_dt_b_asin_title_o04_s02?ie=UTF8&psc=1	$15.99	$47.97
Arduino Nano	A000005	1	https://www.amazon.com/Arduino-A000005-ARDUINO-Nano/dp/B0097AU5OU/ref=sr_1_4?crid=2NESX7F00F9MI&dchild=1&keywords=arduino+nano&qid=1614966616&s=electronics&sprefix=ardu%2Celectronics%2C246&sr=1-4	$20.69	$20.69
Solid Core Wire	22 AWG	1 Kit	Already on hand (price per foot)	$0.09	$5.04
Printed Circuit Board	Custom	1	allpcb.com	$11.00	$11.00
Total					$132.54

Figure 8: Tribble Bill of Materials:

Above is a spreadsheet of materials used for project construction. Some products were purchased in bulk but the prices reflect only items that were utilized. A link to a Bill of materials with clickable links can be found on the showcase website home page.

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.

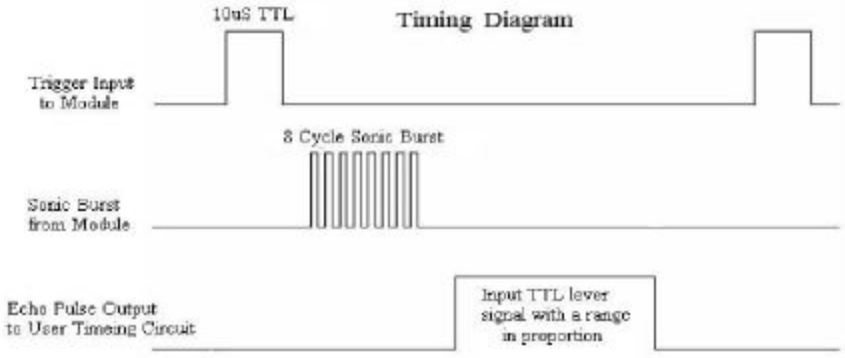

Figure 9: Ultrasonic Distance Sensor HC-SR04 Manufacturer Data

This is the datasheet provided by the manufacturer. The module contains four pins 5V VCC, Trig, Echo, and GND. To measure distance with the module the first step is to send a high pulse to the Trig pin. The module then internally performs a sonic burst and returns a high pulse to the Echo pin. The length of the pulse on the Echo pin corresponds to the distance of the sensor from the nearest object.

Current top-level block diagram (all blocks in the system):

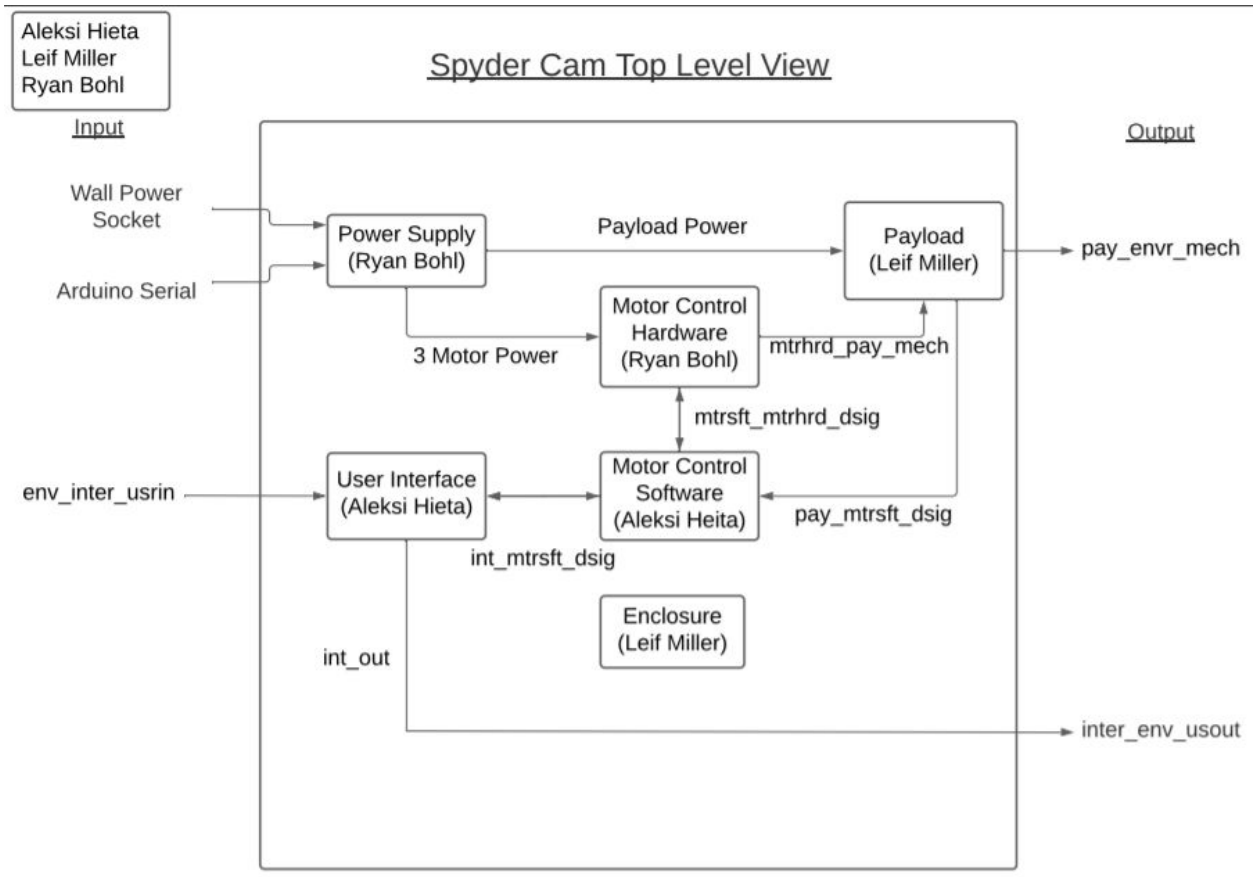

Figure 10: Top level Block Diagram of the Tribble

The tribble is composed of 6 unique design blocks. The enclosure for the system houses the Motors, PCB, Paper, pylons, and pulleys. The payload block is connected to 3 wires which are each moved by the motors and pulleys. The Payload holds various interchangeable writing utensils, as well as a depth sensor to monitor how far away it is from the paper. The Power supply block that ensures the motors and depth sensor receive the necessary power. The motor control hardware block is necessary for driving the motors at the correct speeds and directions that are determined by the software. The user interface is where users input what they wish to draw. The motor control software block determines how the motors will be controlled based on the user input in the GUI.

Current complete interfaces and properties (entire system):

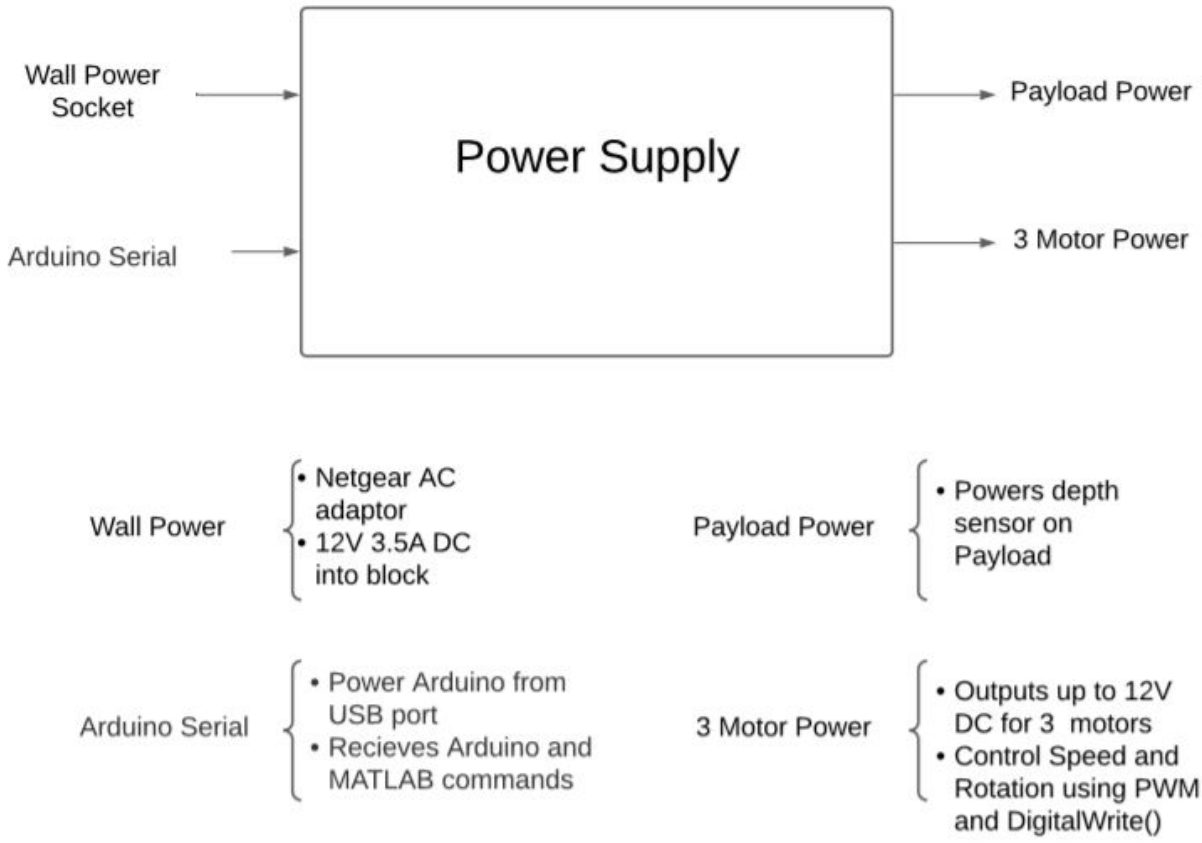

Figure 11: Power Supply interface definitions

There are two inputs to the power supply portion of the system. There is a specific Netgear AC adaptor which converts the 120 volts of AC power coming from the wall socket in your homes into 12 volts, 3.5 Amp DC power that is supplied to the motor drivers. There is also a mini USB serial connection which powers the Arduino Nano in the PCB. This in turn in turn supplies 5 volts to the depth sensor on the payload, as well as a necessary second voltage input for the motor drivers. The 5 volts also power the motor encoders, which display the instantaneous speed of each motor. On most Laptops, the USB serial port supplies 5 volts at 0.5 Amps. Exiting the block is the 5V supplied to the depth sensor, as well as the the speed and rotation signals for each motor, which are determined by the 12V input and the arduino pins.

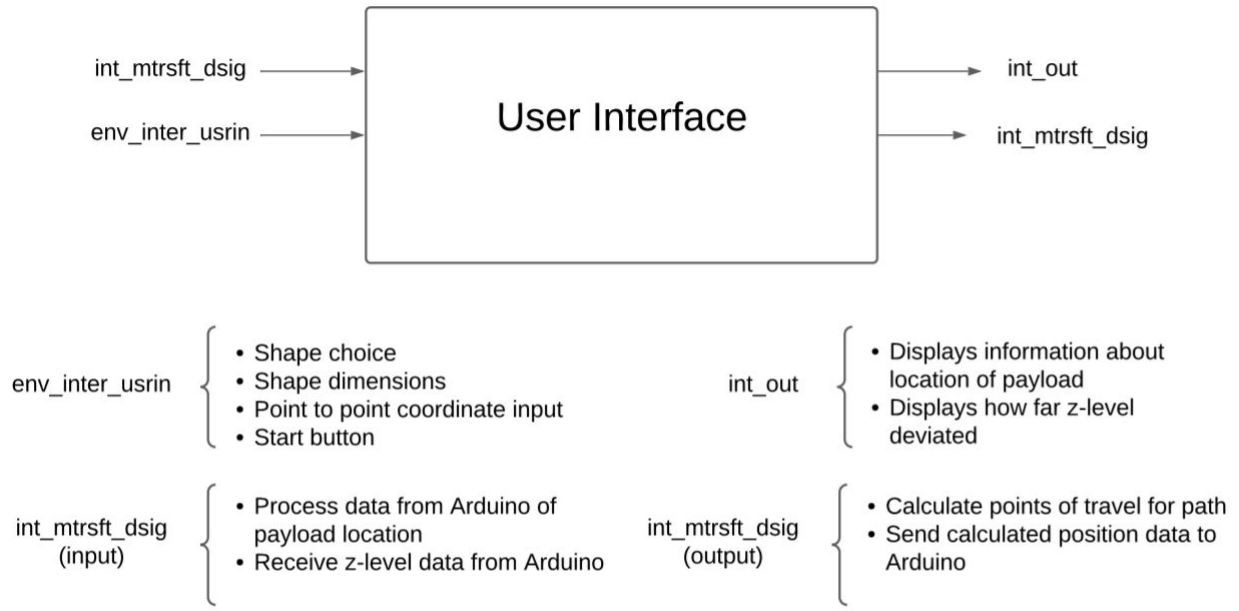

Figure 12: User Interface Definitions

The User Interface consists of the full GUI screen from MATLAB. G-Code commands can be entered into a textbox in the lower left corner as a typical string. From here the inputs are parsed and graphed with plotting data sent to the Arduino. This plotting data consists of the start and end points of the desired line and speeds are calculated to be written to the Arduino pins. Shapes chosen utilize the same graphing function to ensure proper lines are drawn. Additionally, the current position of the payload is shown to the right of the inputs to inform the user what the path of travel is. This way, the User Interface verifies that the correct commands are inputted from the user, which aligns with the real world movement.

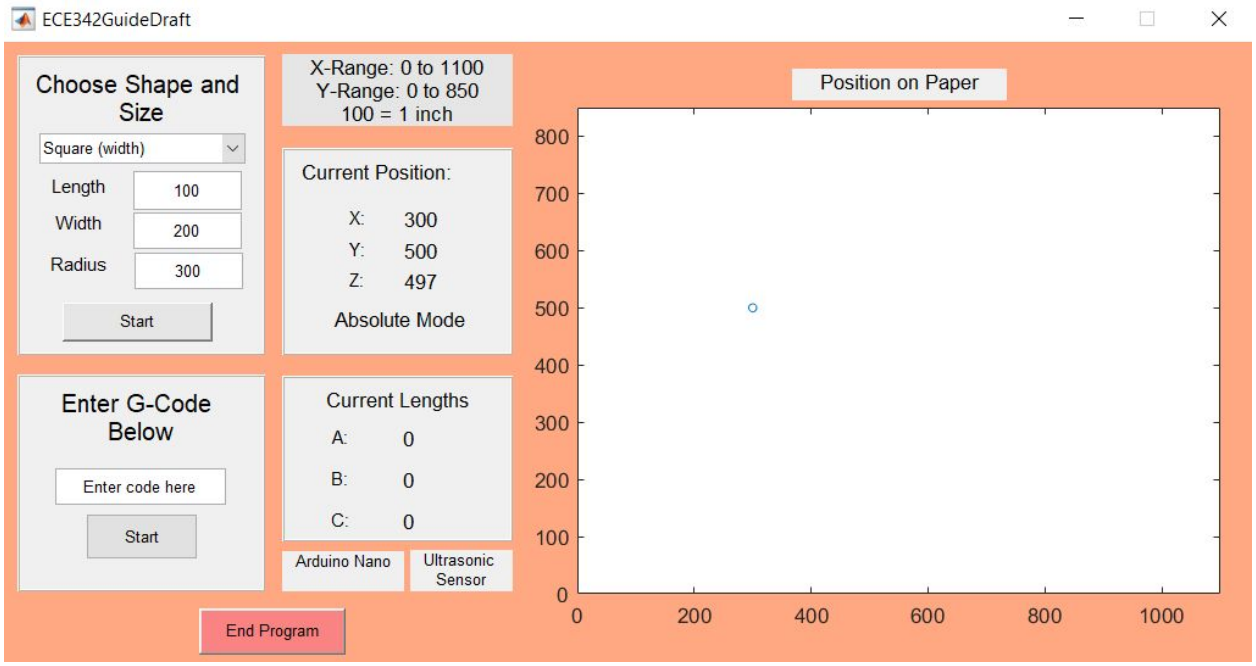

Figure 13: Completed GUI

The MATLAB GUI has evolved over the course of the project from beginning with only the G-Code portion to all the current elements depicted. The Current Lengths box shows the lengths of wire for each of the motors to the payload. Additionally, the X, Y, and Z positions are based on the drawing instrument relative to the paper. As shown on the graph, the 8.5 x 11 piece of paper is put into graphical axes by 100th of an inch.

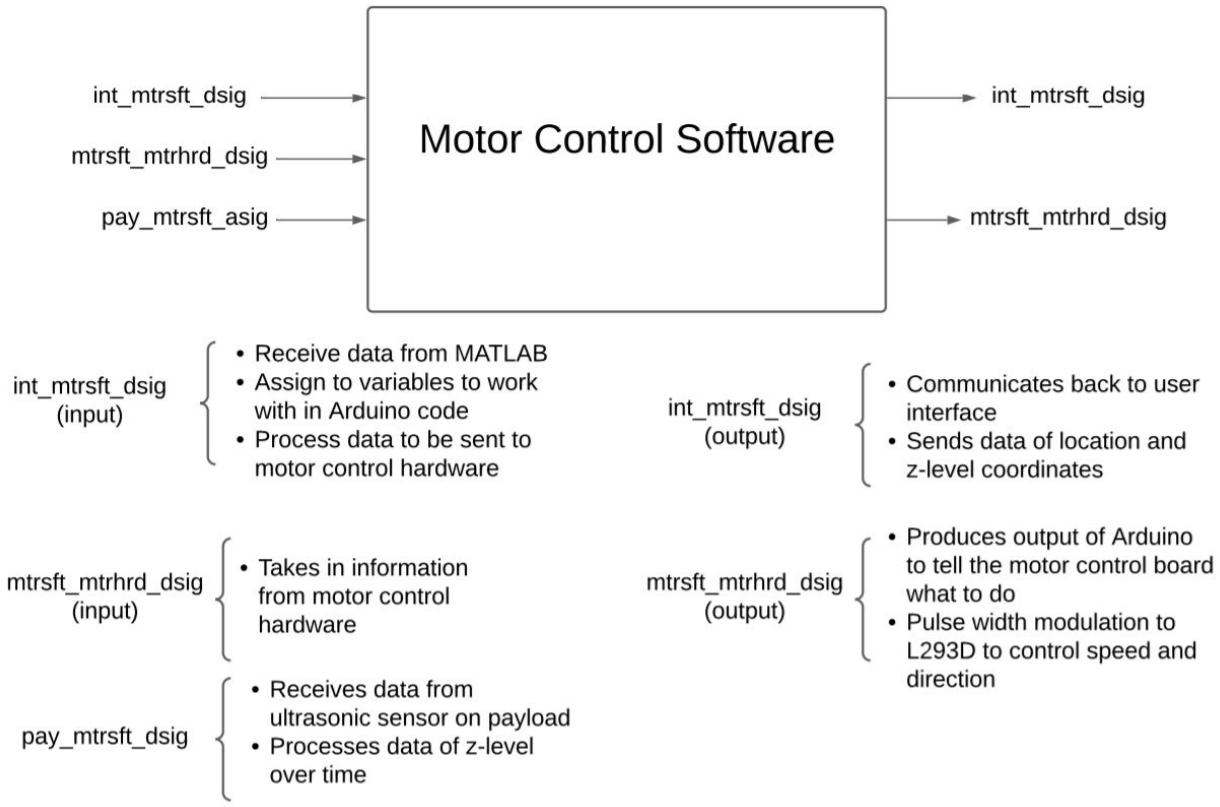

Figure 14: Motor Control Software Interface Definitions

The Motor Control Software is responsible for writing values to the Arduino. From the Arduino digital pins, signals are sent to the L293D boards to control the motors. The graph_function.m file is the main software file for converting data to movement. This file takes point data from the GUI to calculate the length of each wire along the XY line, and in turn the speeds through the change in distance between points. Speeds are adjusted based on the performance of individual motors, added tension on release, and overall distances to travel. The up_down.m file also aids in ensuring the payload descends to the proper height through reading the ultrasonic sensor. Values from the ultrasonic sensor are also read through the graphing function back to the GUI to inform the user of the current height of the payload.

```
function graph_function(a, u, xprev, yprev, xval, yval)
        F1 = 'D2';
        B1 = 'D4';
        F2 = 'D7';
        B2 = 'D8';
        F3 = 'D11';
        B3 = 'D12';

        E1 = 'D3';
        E2 = 'D6';
        E3 = 'D10';

        %Fwd and Bck PWMVoltage Range: 0 - 5
        %Minimum rotation voltage ~0.8

        distance = sqrt((xval-xprev)^2+(yval-yprev)^2)
        n = ceil(distance/100)
        xline = linspace(xprev,xval,n);
        yline = linspace(yprev,yval,n);

        Abase = [];
        Bbase = [];
        Cbase = [];
        Awire = [];
        Bwire = [];
        Cwire = [];

        for i = 1:1:n
            Abase = [Abase, floor(sqrt((650+xline(i))^2+(238+yline(i))^2))];
            Bbase = [Bbase, floor(sqrt((550-xline(i))^2+(1938-yline(i))^2))];
            Cbase = [Cbase, floor(sqrt((1850-xline(i))^2+(238+yline(i))^2))];

            Awire = [Awire, floor(sqrt((Abase(i))^2+(900)^2))];
            Bwire = [Bwire, floor(sqrt((Bbase(i))^2+(900)^2))];
            Cwire = [Cwire, floor(sqrt((Cbase(i))^2+(900)^2))];
        end

        Aspeed = [];
        Bspeed = [];
        Cspeed = [];

        %========ADJUST FOR PROPER SCALING OF CHANGE IN=================
        %=============DISTANCE TO SPEED================================
        timeconstant = 30.0; %Adjust for speed differences
        for i = 1:1:n-1
            Aspeed = [Aspeed, (Awire(i+1)-Awire(i))/timeconstant];
            Bspeed = [Bspeed, (Bwire(i+1)-Bwire(i))/timeconstant];
            Cspeed = [Cspeed, (Cwire(i+1)-Cwire(i))/timeconstant];
        end

        for i = 1:1:n-1
            if Aspeed(i) > 5
                Aspeed(i) = 5;
            end
```

```
            if Aspeed(i) < -5
                Aspeed(i) = -5;
            end
            if Aspeed(i) > 5
                Aspeed(i) = 5;
            end
            if Bspeed(i) < -5
                Bspeed(i) = -5;
            end
            if Bspeed(i) > 5
                Bspeed(i) = 5;
            end
            if Cspeed(i) < -5
                Cspeed(i) = -5;
            end
            if Cspeed(i) > 5
                Cspeed(i) = 5;
            end
        end

        tic;
        height = readDistance(u)*38
        for i = 1:1:(n-1)
            if(Aspeed(i) > 0)
                writePWMVoltage(a, E1, 0.6*Aspeed(i)); %Loosen
                writeDigitalPin(a, F1, 1);
                writeDigitalPin(a, B1, 0);
            else
                writePWMVoltage(a, E1, -(0.84)*Aspeed(i)); %Tighten
                writeDigitalPin(a, F1, 0);
                writeDigitalPin(a, B1, 1);
            end
```

```
            if(Bspeed(i) > 0)
                writePWMVoltage(a, E2, (0.8)*Bspeed(i)); %Loosen
                writeDigitalPin(a, F2, 1);
                writeDigitalPin(a, B2, 0);
            else
                writePWMVoltage(a, E2, (1)*-Bspeed(i)); %Tighten
                writeDigitalPin(a, F2, 0);
                writeDigitalPin(a, B2, 1);
            end
            if(Cspeed(i) > 0)
                writePWMVoltage(a, E3, (0.82)*Cspeed(i)); %Loosen
                writeDigitalPin(a, F3, 1);
                writeDigitalPin(a, B3, 0);
            else
                writePWMVoltage(a, E3, (1)*-Cspeed(i)); %Tighten
                writeDigitalPin(a, F3, 0);
                writeDigitalPin(a, B3, 1);
            end

            curr_height = readDistance(u)*38;

        end
        toc
        writeDigitalPin(a, F1, 0);
        writeDigitalPin(a, B1, 0);
        writeDigitalPin(a, F2, 0);
        writeDigitalPin(a, B2, 0);
        writeDigitalPin(a, F3, 0);
        writeDigitalPin(a, B3, 0);
```

Figure 15: MATLAB graph_function.m code

Graphing_Function.m for Motor Software takes in the starting X, Y and ending X, Y. From here, the distances are calculated. Next, the speeds of each wire changing over the course of the line makes three arrays of speed values ranging from -5 to 5. Negative values indicate movement in the opposite direction of positive values. The coefficients also help to ensure that the motors spin at correct speeds despite performance variation.

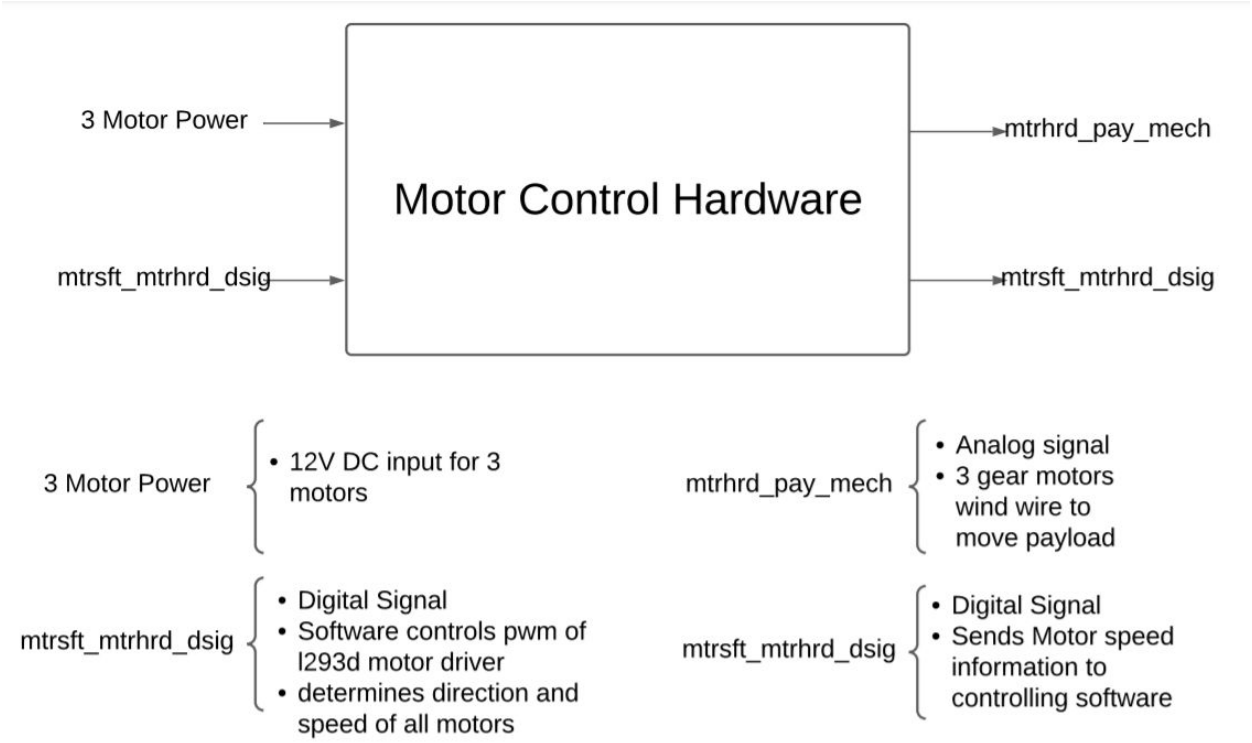

Figure 16: Motor Control Hardware Interface Definitions

The motor control hardware block takes 12 volts from the power supply to power the l293d motor drivers, allowing the motors to receive up to 12 volts. There are also 9 digital signals that are provided by the arduino--a PWM speed signal, clockwise direction signal, and a counter clockwise direction signal for each driver. Exiting the block is the mechanical winding of the wire for moving the payload. There are also encoder signals that show the speed of each rotation signal sent to the motor. These 6 signals can be read by the same arduino for increasing the accuracy of the motor control, although this was not implemented in the final design of the Tribble.

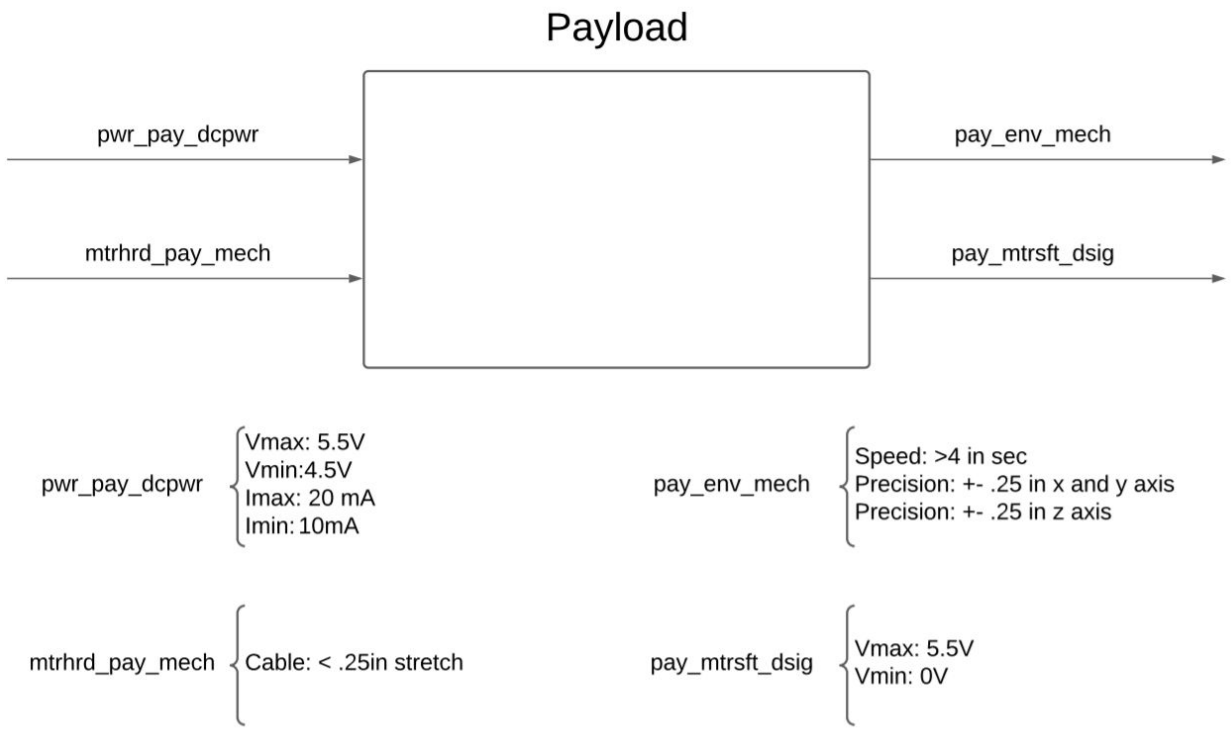

Figure 17: Payload Interface Definitions

The payload block is the portion of the system that houses the writing utensil and is moved by the system to create the desired outcome. The ultrasonic distance sensor is attached to the payload and thus the payload requires power and communication connections. 5V DC power is needed for the sensor and comes from the 5V node on the Arduino Nano. The Trig and Echo Pins of the sensor require arduino I/O pins for driving and reading. The interface between the payload and the motors that are attached to the enclosure needs to be of known length for calculation accuracy.

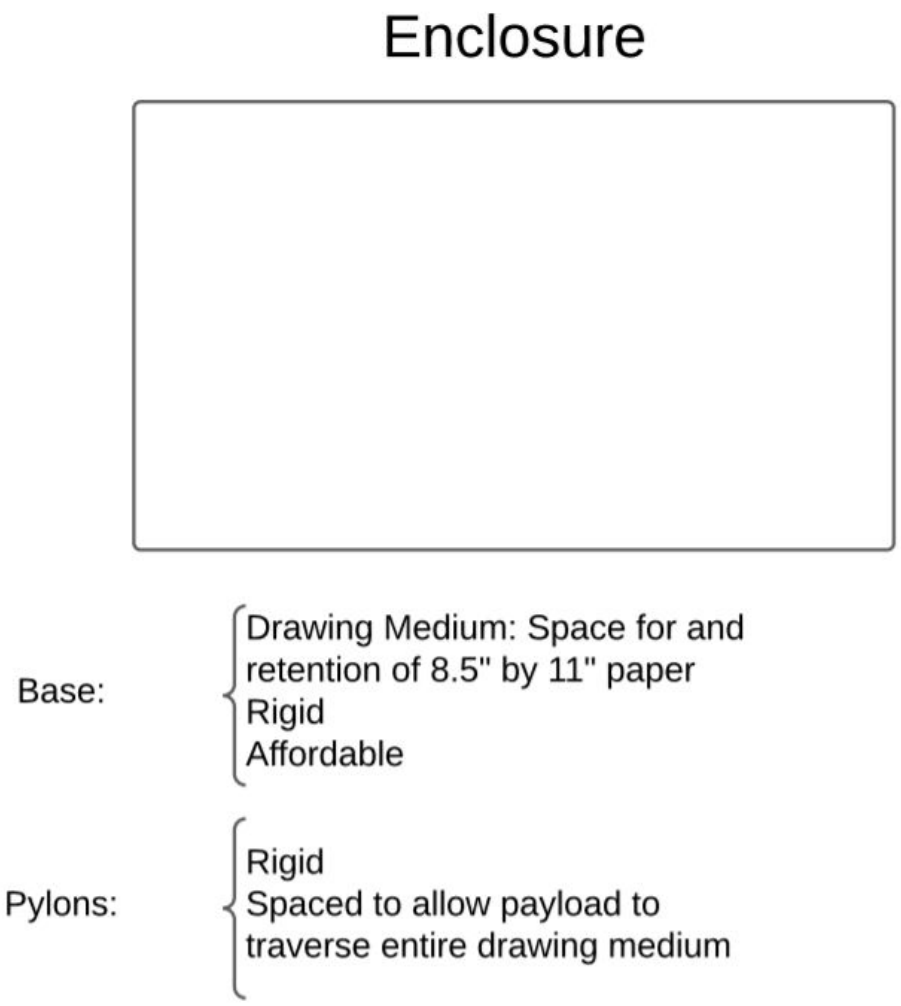

Figure 18: Enclosure Interface Definitions

The enclosure block is the bulk of the physical product. The enclosure is designed to retain the piece of paper and provide attachment points for all system hardware. System hardware includes the motors and the motor control pcb. The enclosure also defines the payloads field of operation thus the enclosure must be large enough to allow the payload to position above any point on the piece of paper.

PCB Information:

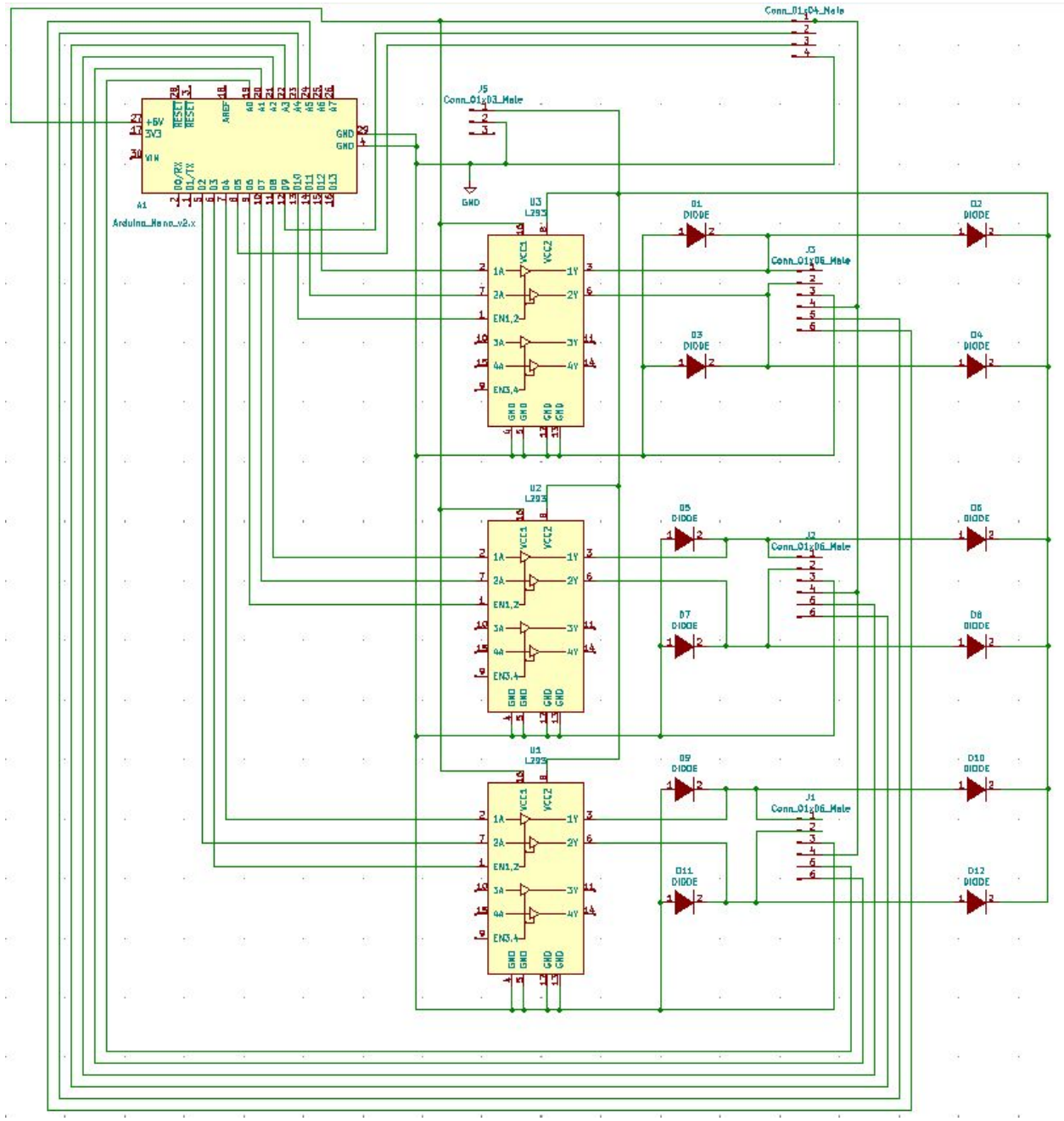

Figure 19: Full PCB Wiring Schematic

Figure 19 shows the schematic for the PCB implemented in the project. The PCB combines the power supply block and the motor control hardware block onto a single board. The top left of the schematic shows the Arduino Nano which outputs the speed and direction signal to each of the motor drivers. Although a single motor driver can drive two motors, three drivers were used in the design to reduce clutter on the PCB. The Nano outputs 5 volts which are used by the motor drivers, motor encoders, and the depth sensor (which has a connector in the top right of the schematic). The Nano has two connections for configuring the depth sensor and reading its values. Finally, on the top of the Nano there are 6 analog inputs which are used for reading the motor encoder values. Above the motor drivers, there is the pin connection for the 12V barrel jack input. All of the diodes in the schematic work to clamp the voltages supplied to the motors between ground and 12 volts. All elements (besides 6 of the diodes on the right) share a common ground.

PCB:

The PCB is 4 by 5.15 Inches. It contains three mounting holes so it can easily be attached to the wooden enclosure. All components are through-hole. J1 through J4 should be male pin headers. J5 is the barrel jack. A1 is for the arduino nano--it is recommended to solder female pin headers here. The arduino should be orientated with its USB pointing out of the left side of the PCB. A completely assembled and connected PCB is depicted in Figure 22.

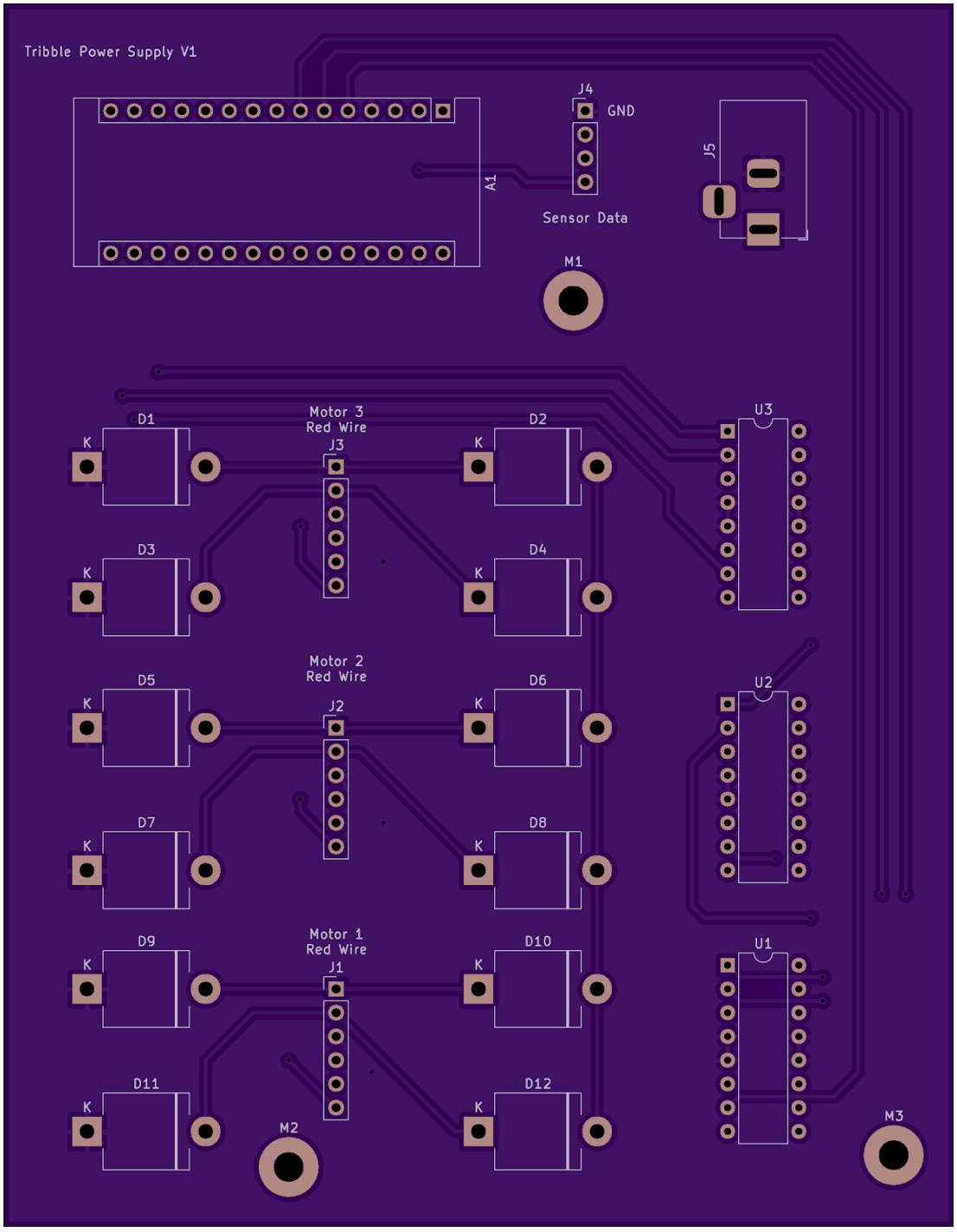

Figure 20: Top Layer of Bare PCB

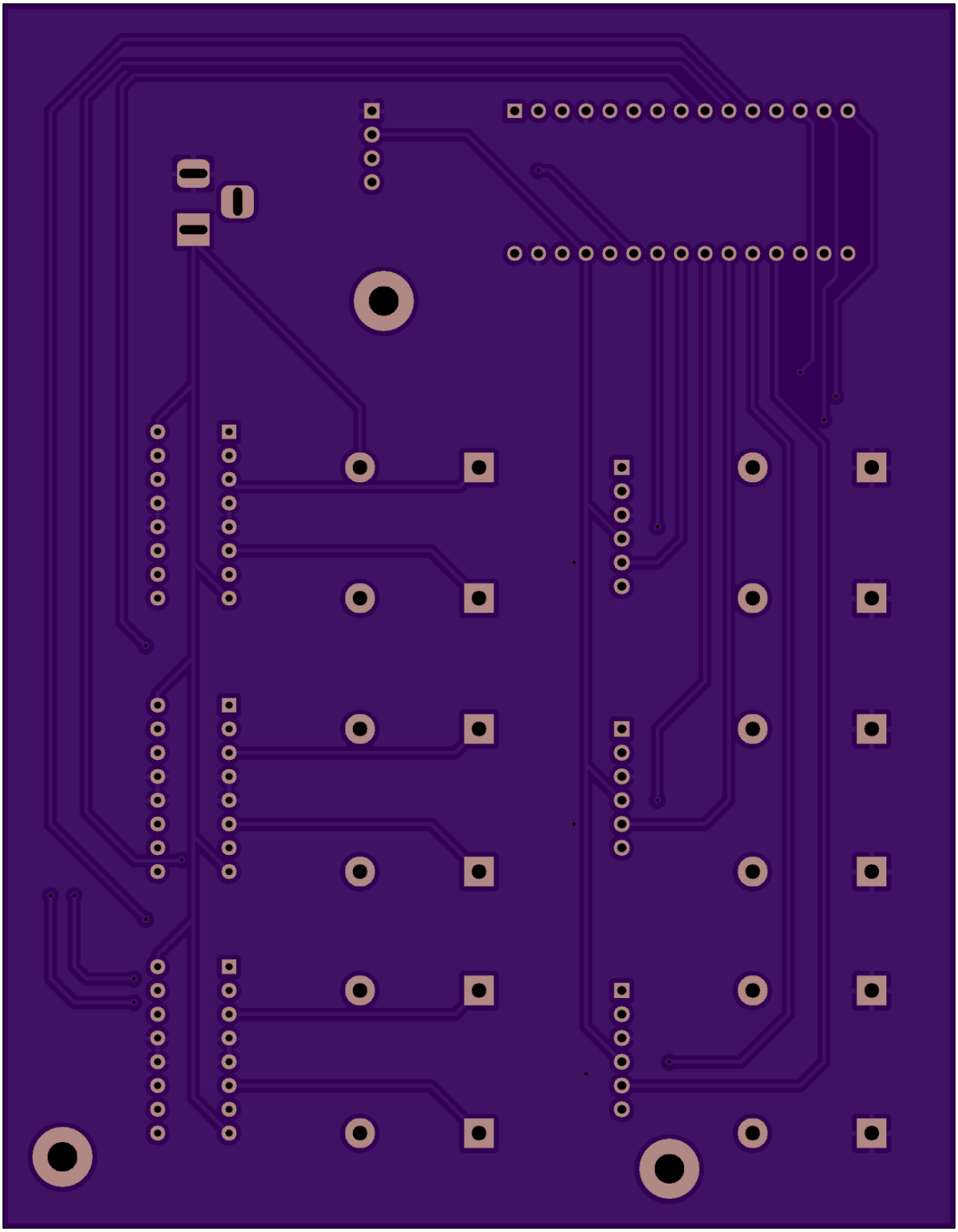

Figure 21: Bottom Layer of Bare PCB

Figure 21 is the bottom side of the bare PCB used in the Tribble. There are modifications that must be made to have in order to have the PCB function as depicted in the schematic. Both VCCs are connected to 12V as seen with the traces that connect to the barrel jack. To get around this, 5 volts needs to be pulled from the from the depth sensor pin and connected to each of the VCC1 inputs. Make sure that there is no connection of the 5V rail and the 12V rail or else it will destroy the nano and depth sensor if they are connected. There also needs to be a modification to the arduino nano so that pins d5 and d9 can connect to the depth sensor. Both of these modifications can be seen in Figure 22. A design revision would be useful so that manual modifications do not have to be made to the PCB.

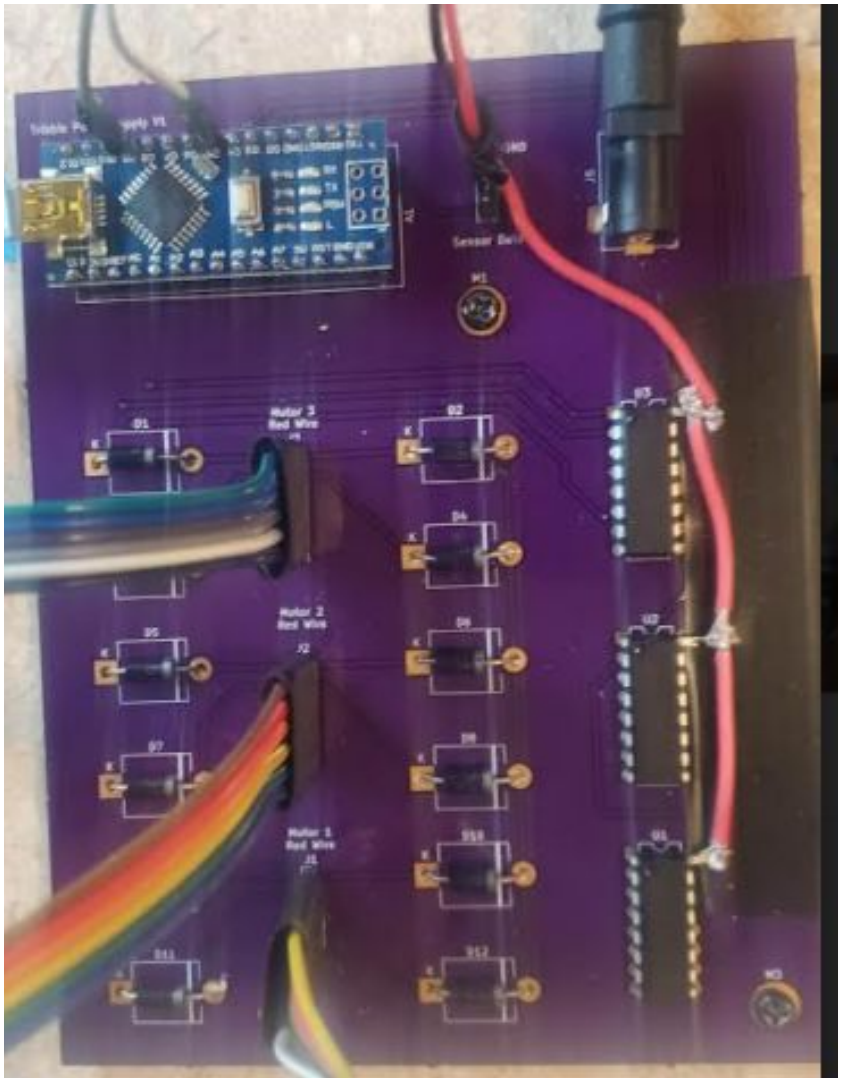

Figure 22: Connected PCB with Modifications