Daily Log

Sunday, June 16, 2024 7:25 PM

6/10

- Cut router table 2 x 4s
- 

6/11

- Assembled router table 2 x 4 frame

6/12

- Reduced table height by 4 inches and added side panels
- Cut top work surface

6/13

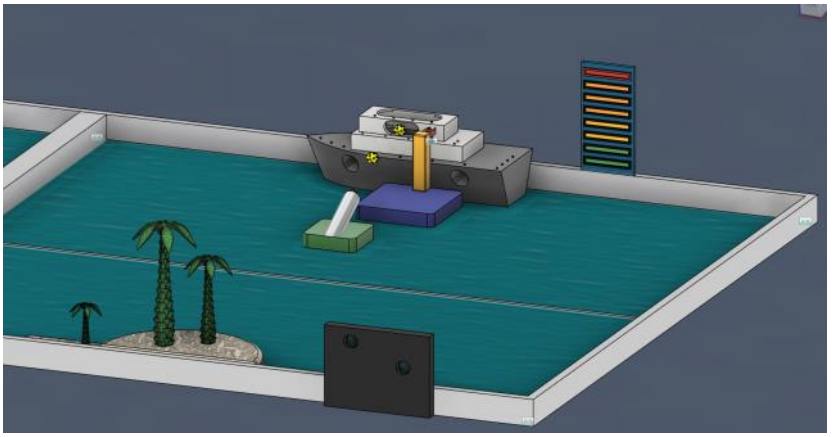
- Installed top, bottom, and back panels
- Added wheels
- First 2 coats of paint

6/14

- Last 2 coats of paint on router table

6/15

- Completed first iteration of game design
    - Initial CAD design of game
    - First iteration of rules and point system

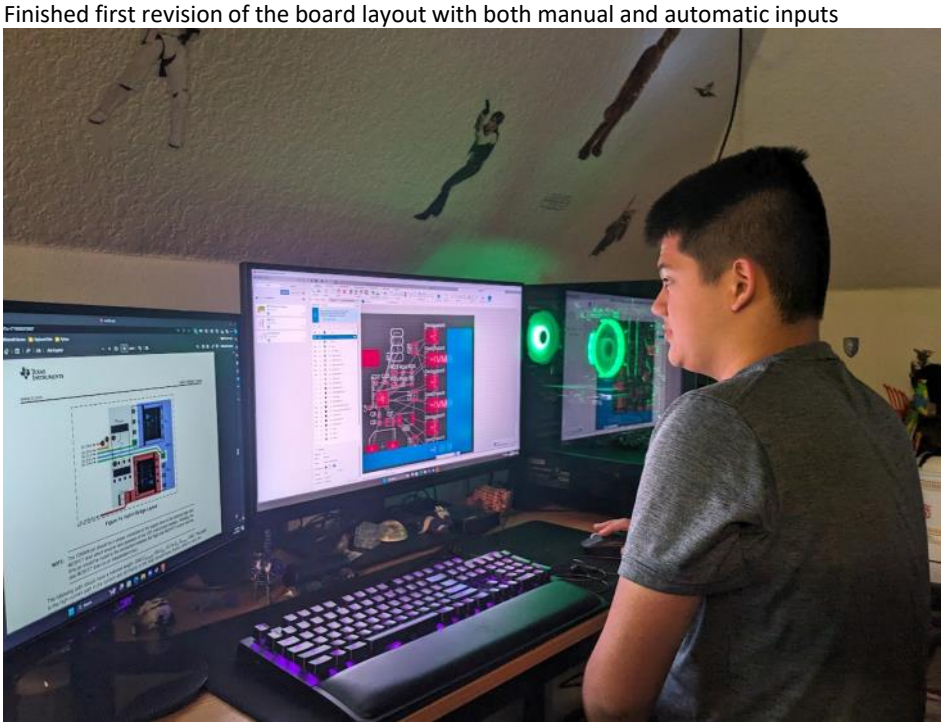
- 

6/16

- Created schematic for DRV8320 with parts selected from JLCPCB

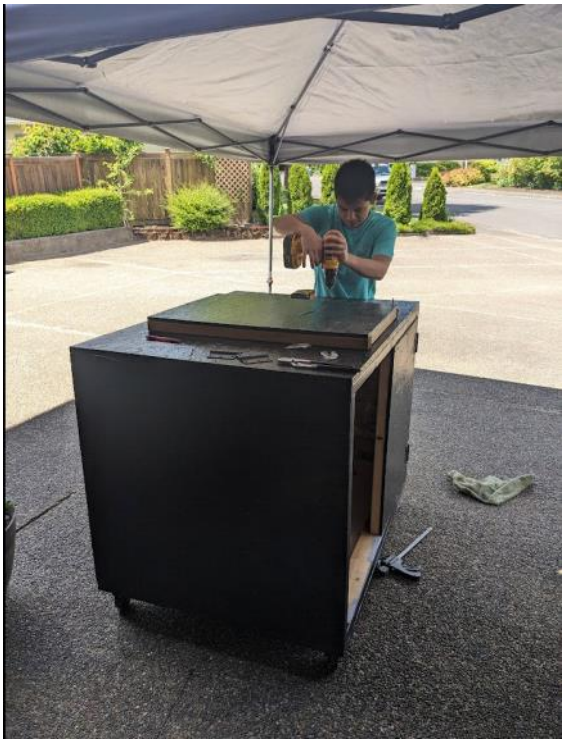
6/17

- ~ 4 hours
- Finished first rev of schematic
- Finished first revision of the board layout with both manual and automatic inputs



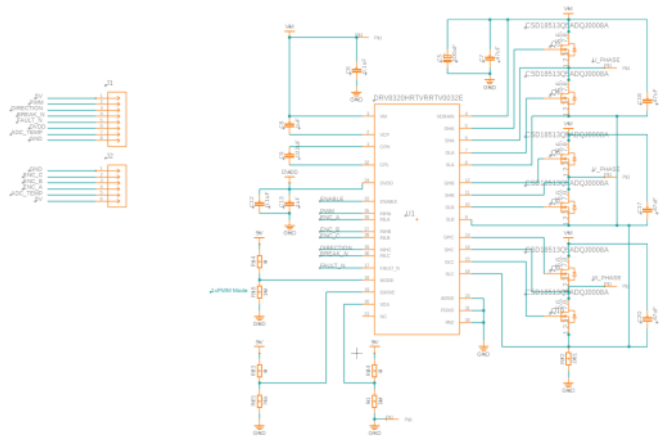
6/18

- ~ 7 hours
- Added doors to the router table
- Assembled the router
- Added PCB connectors and started looking at JLCPCB assembly requirements



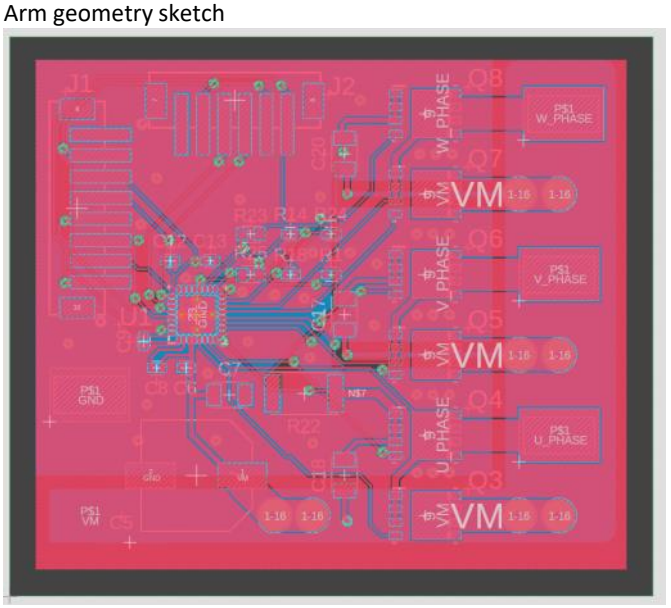
6/19

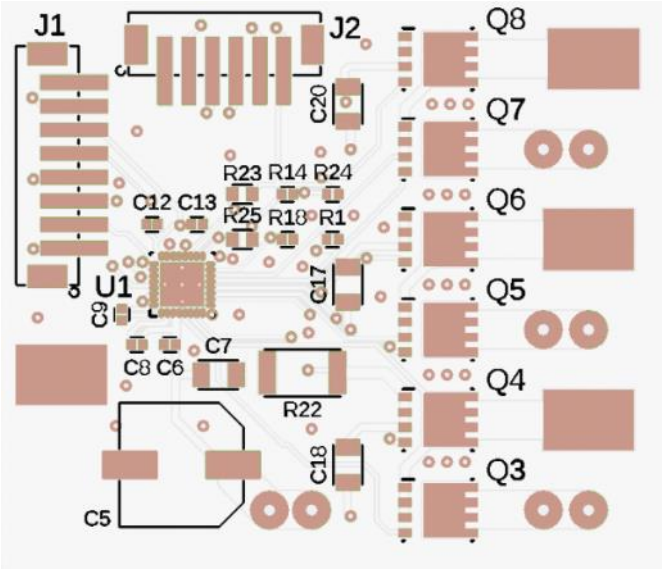
- ~ 4 hours
- CAM day 8 CADvent for the first router test
- Finished the first revision of the motor controller PCB

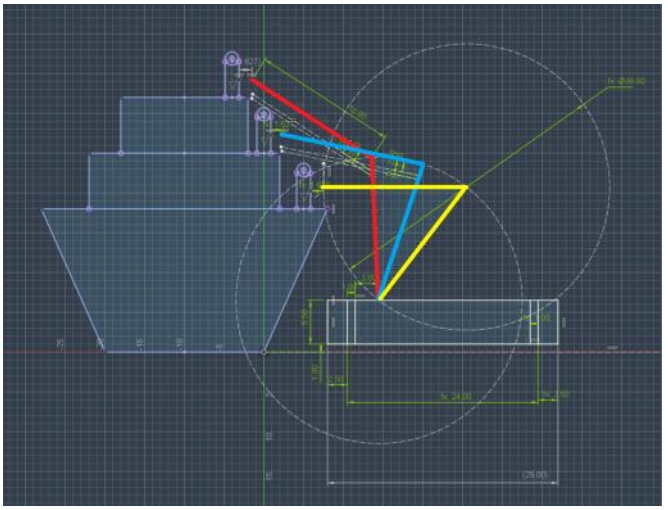


6/20

- ~ 5 hours
- Fixed PCB based off of visual inspection and JLCPCB DFM tool
- Submitted motor controller PCB to JLCPCB
- Brainstorming robot designs for game
- Arm geometry sketch



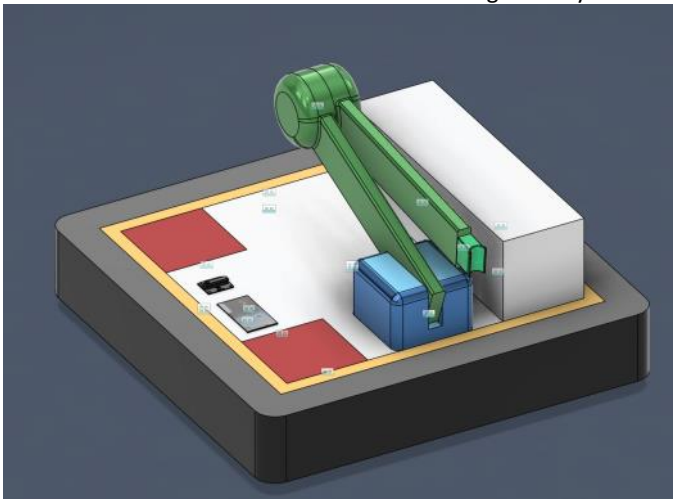


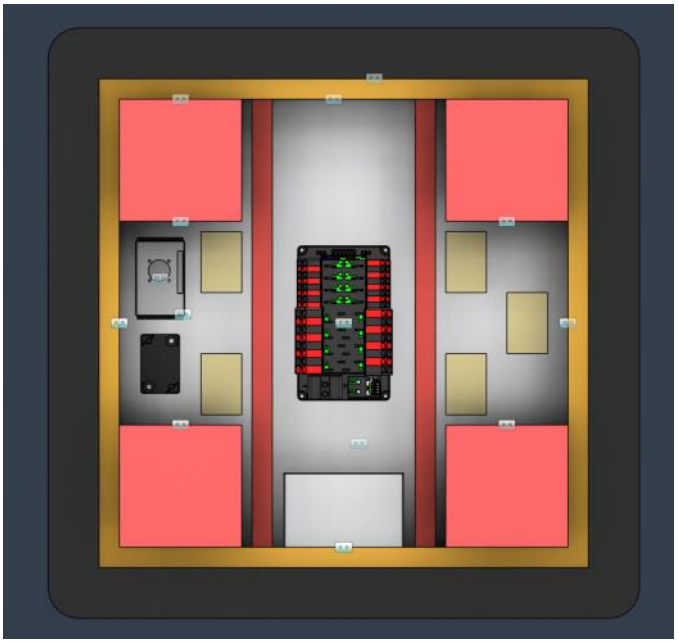


6/21

~ 3 hours

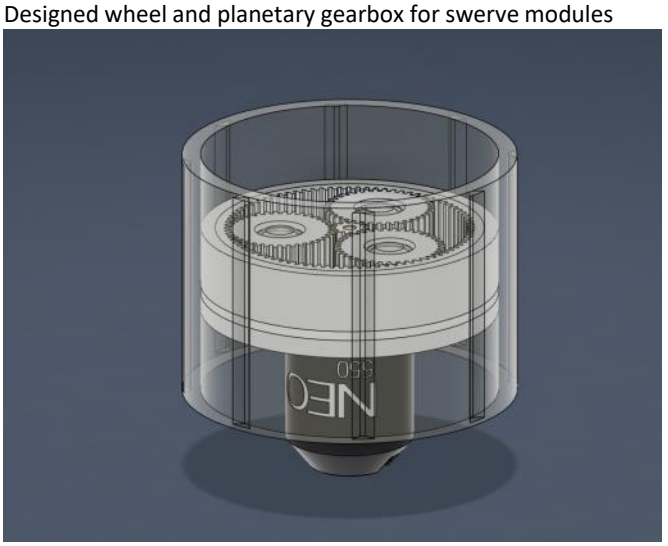
- Constructed an abstract robot based off of the geometry sketches





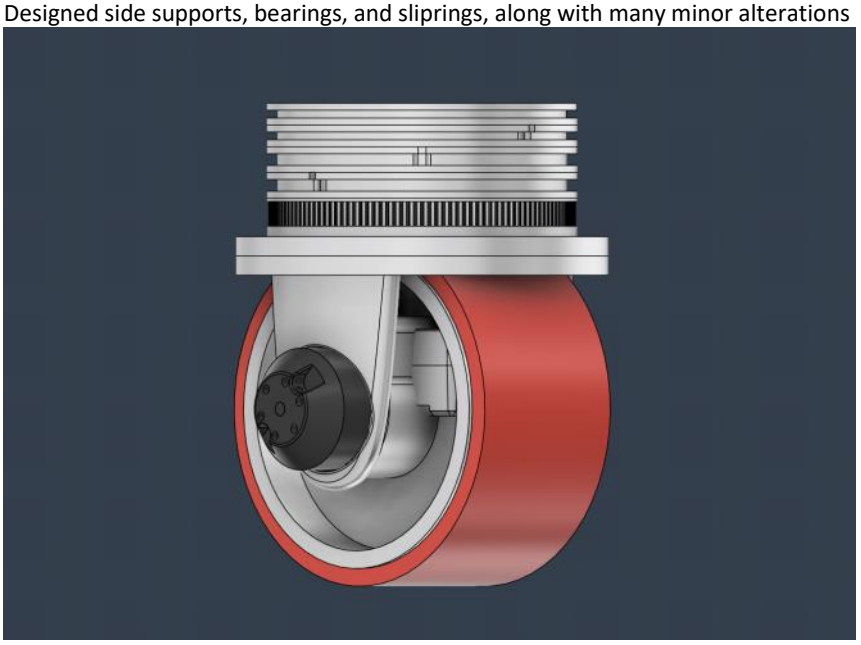
6/22

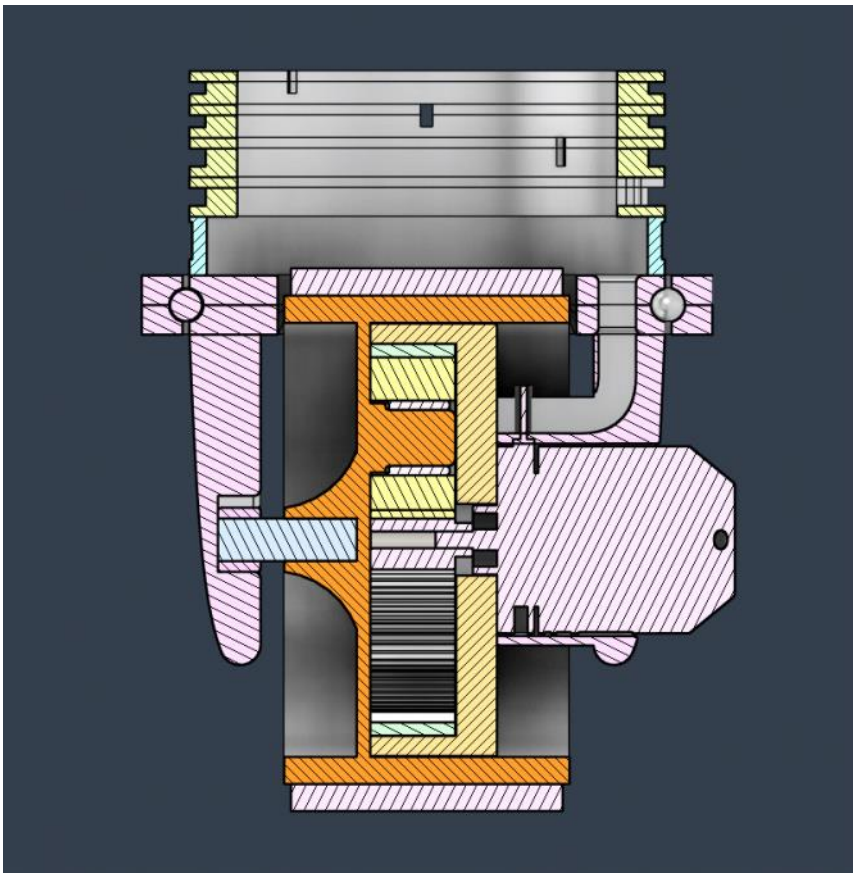
- ~ 5 hours
- Designed wheel and planetary gearbox for swerve modules

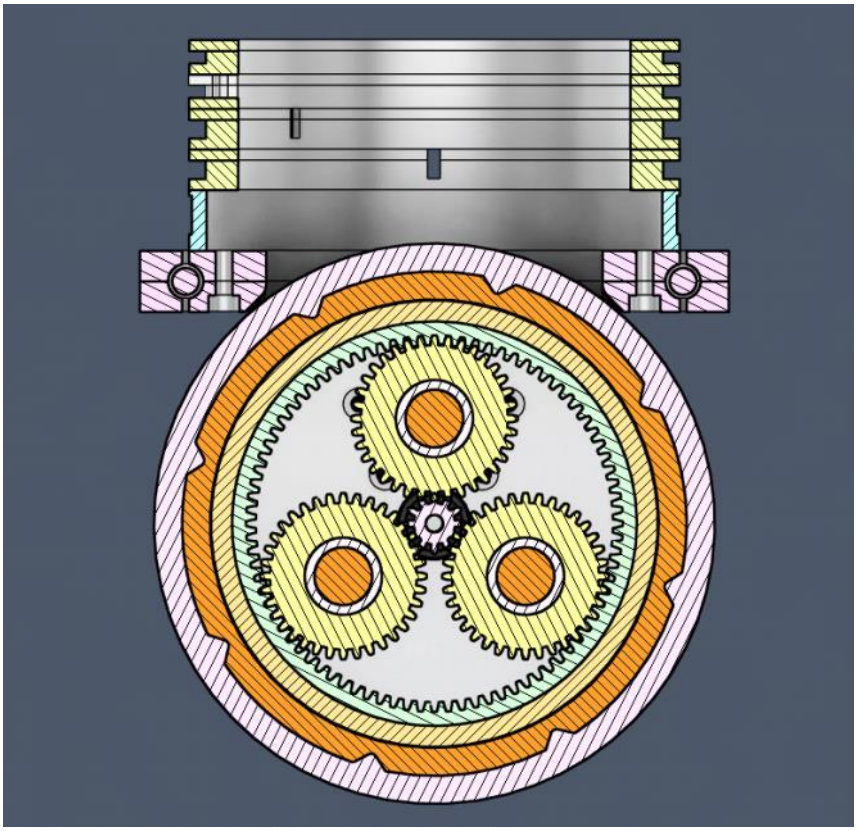


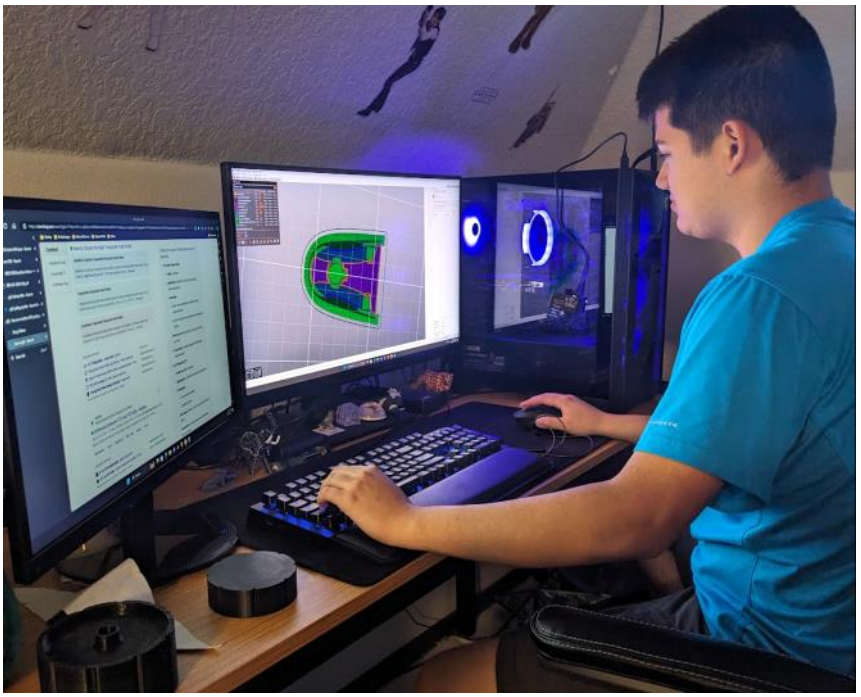
6/24

- ~ 7 hours
- Designed side supports, bearings, and sliprings, along with many minor alterations



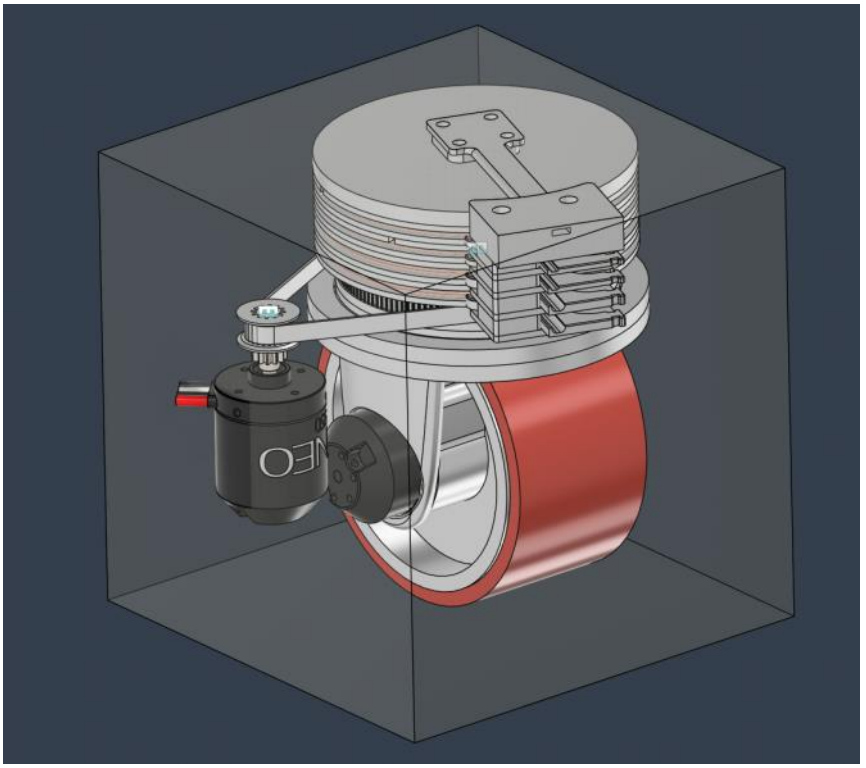


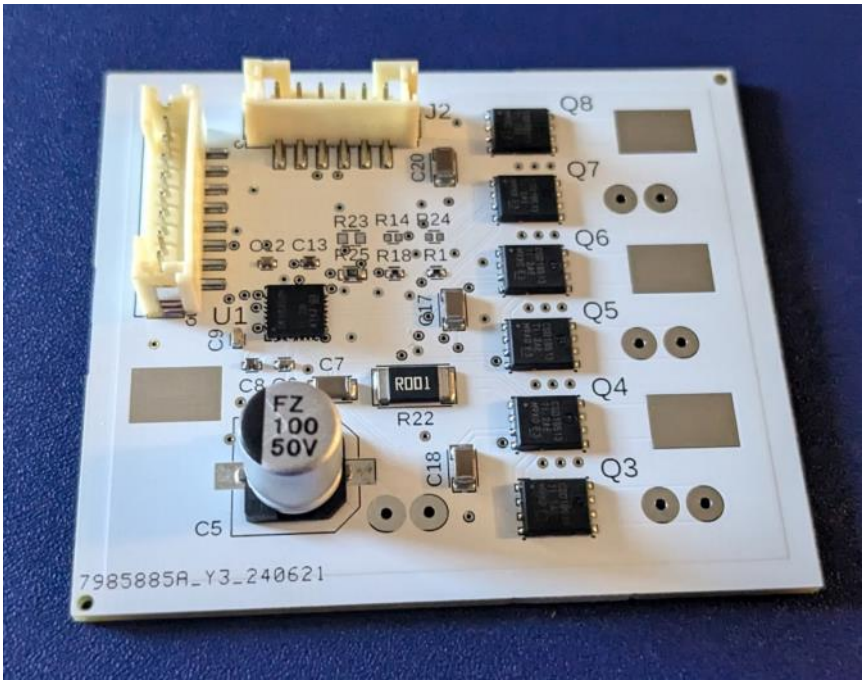




6/25

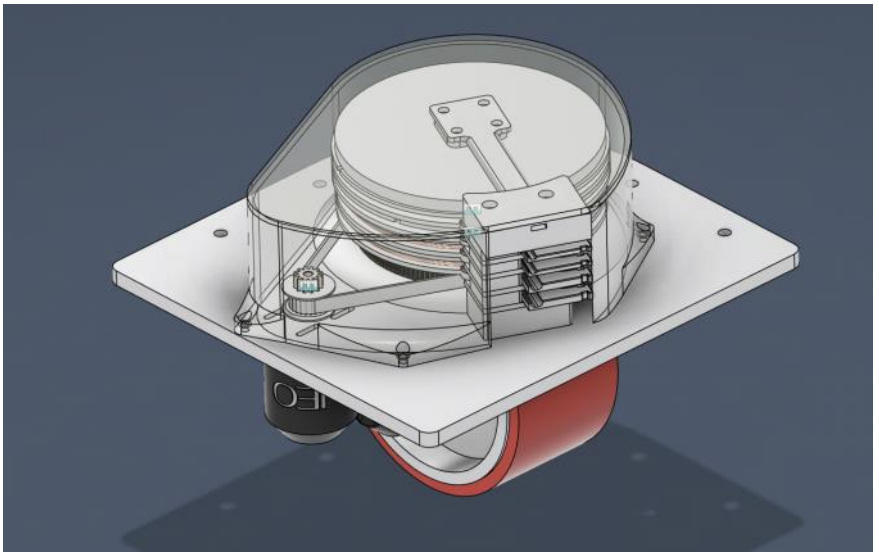
- ~ 7 hours
- Added detail to sliprings and created the slipring brushes
- Created GT2 belt assembly
- The motor controller board arrived





6/26

- ~ 4 hours
- Built the swerve platform and cover
- Made adjustments to the GT2 pulley and slipring tolerances



6/27

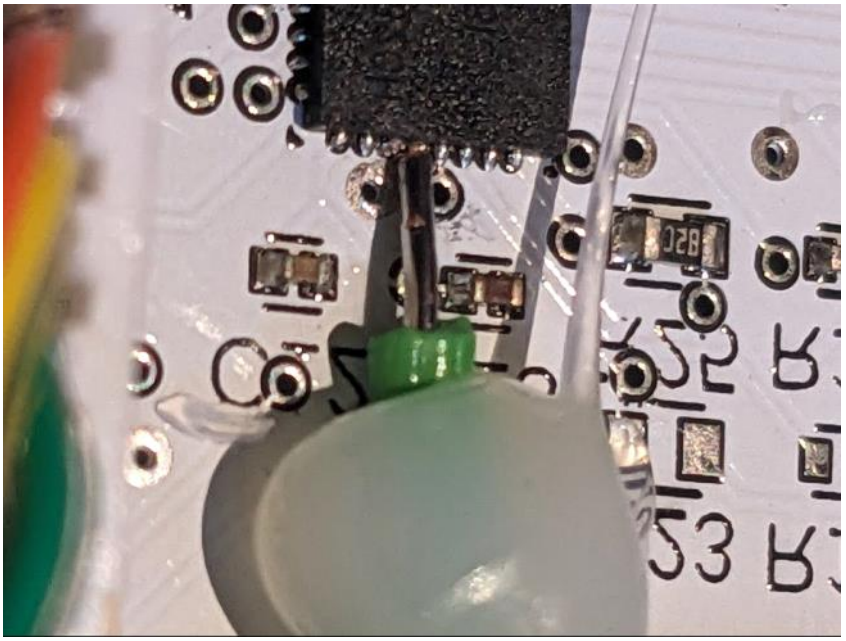
- ~ 4 hours
- Manually crimping JST connectors with needle nose pliers to test motor controllers
- Constructed the sliprings
- Wrote Arduino code for motor controller testing
- Created the corner of the frame

6/28

- ~ 3 hours
- Made minute changes to the top slipring cap, the slipring brushes, and a few other changes to the swerve modules
- Began testing the motor controllers, unfortunately no luck

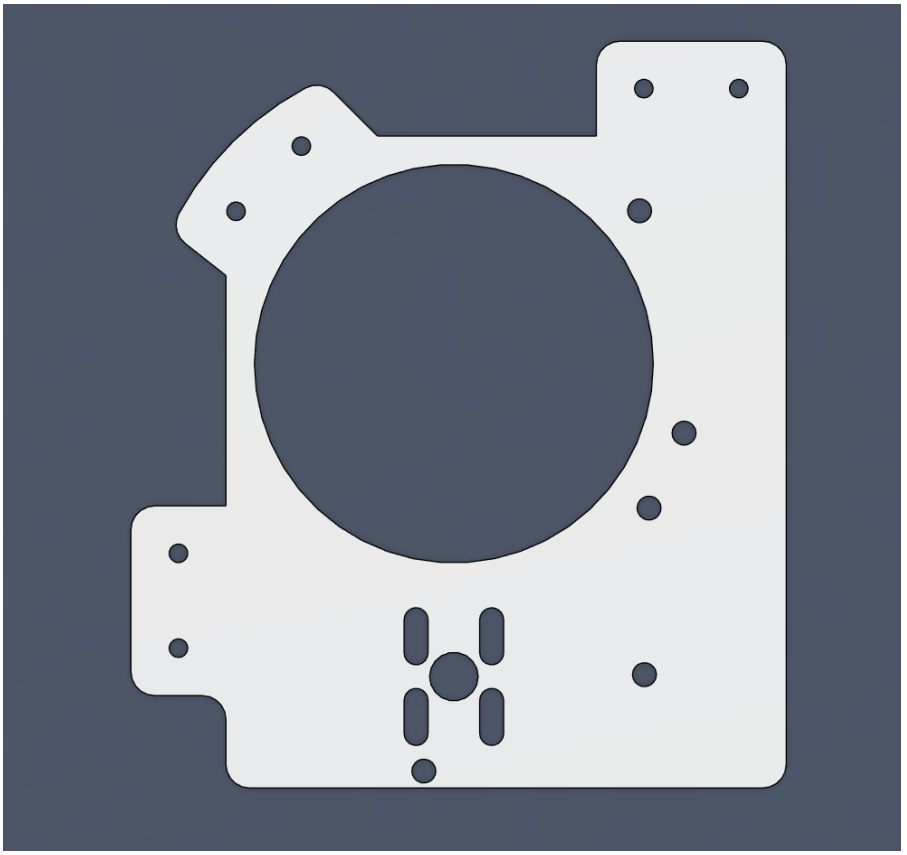
6/29

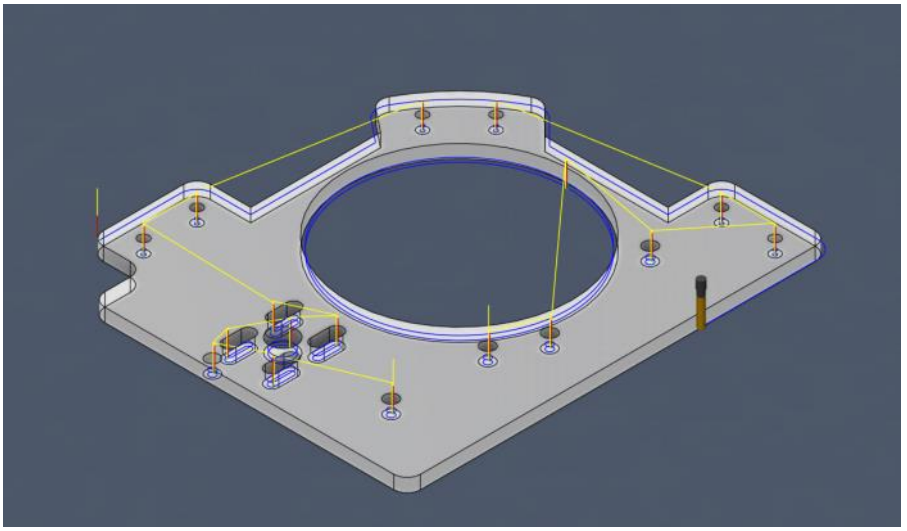
- ~ 3 hours
- Continued to debug the motor controller
    - After reviewing the schematic we found that the DVDD pin had been connected to the connector pin instead of ENABLE.
    - After spending a few hours we managed to connect a wire to the ENABLE pin and connected it back to the Arduino.
    - Unfortunately, the pin next to enable was ground so shorting the pins together was not an option; fortunately, the pin to its right is a NO CONNECT pin which we could short to. This allowed us to wedge the wire between the two pins.



6/30

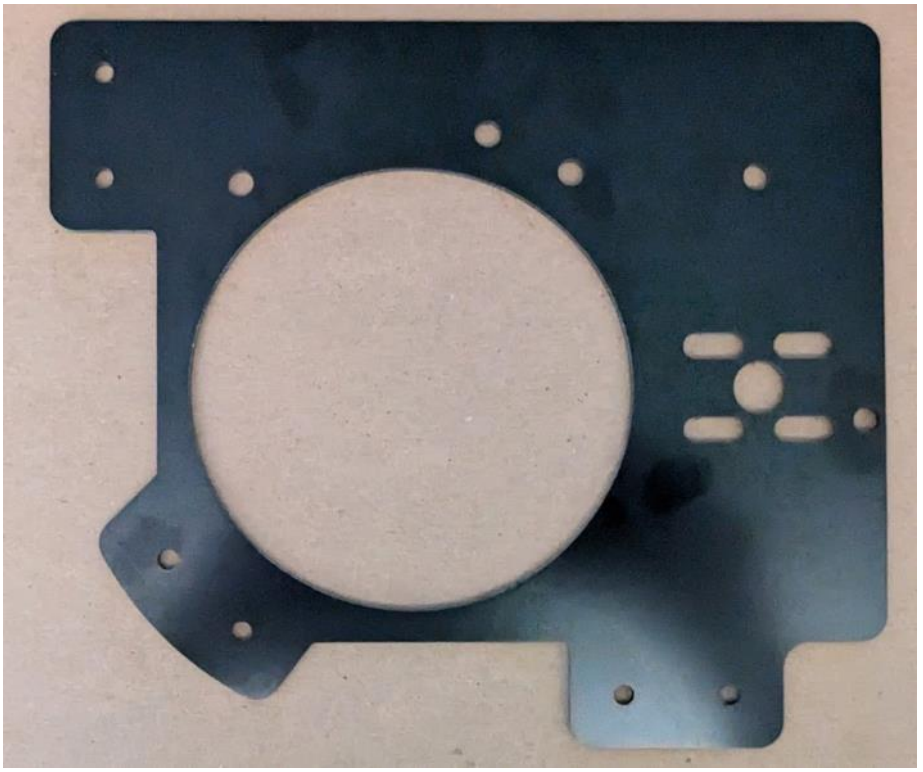
- ~ 5 hours
- Additional motor controller testing to figure out reliability of different wiring techniques (single vs multi-threaded wire, gauge, etc.)
- Designed the chassis bars and chassis connectors, and altered the swerve base plate to fit in via slots.
- 3D printing the swerve base plate was not successful, so we CAMed the plates so that we could cut them on the CNC router
    - Due to the router bit being larger than an M3 hole, we had to increase the size of those holes.





7/1

- ~ 4 hours
- Using the CNC router we cut the base plate for the swerve modules
- We began to assemble the swerve modules, making minor adjustments in CAD when we found issues with tolerances



7/2

- ~ 1.5 hours
- Test fit the plate into the corner chassis and found a few issues with the tabs
- Fixed the tabs in CAD and modified the corner mount to have one robust support instead of two
  To Do
- Adjust power sliprings
- Fix bar inserts and corner

7/3

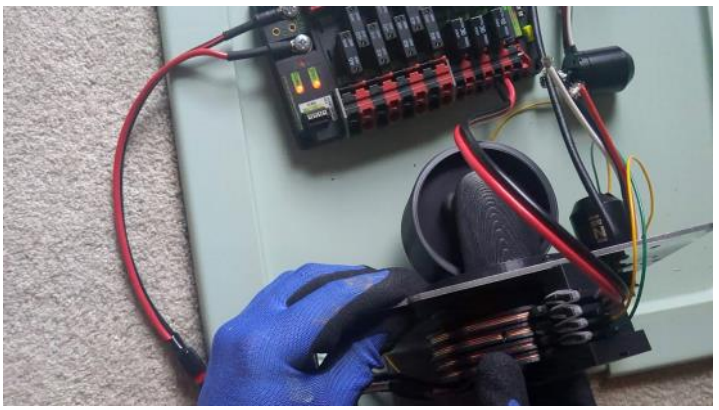
- ~ 4 hours
- Fixed the base plate tabs and re-routered it
- Fixed the chassis bars and corner sizing
- Other tolerance adjustments to the sliprings and side supports
- Began creating the arm in CAD

7/4

- ~ hours

7/5

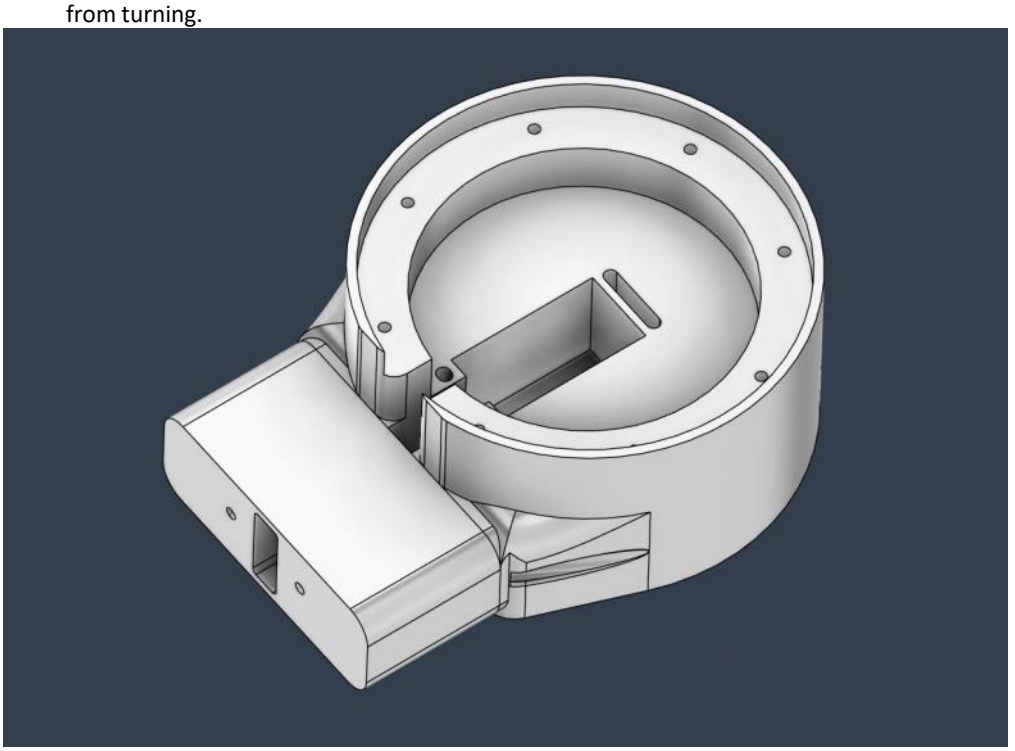
- ~ 5 hours
- Integrated SparkMAX into the slipring assembly. The sliprings were able to power the SparkMAX like normal however the sliprings would slip or get caught due to the uneven pull created by the spring on the one side
- Tested the swerve modules turning with the sliprings; it was successful however the belts slipped more often than we preferred
- The variance created by the 3D printing created tolerance issues in our 3 x 1 bar inserts and the slipring assembly
- In order to find the correct interface for the bar inserts we tested small 3D printed frames to find the correct size that accounted for the 3D printer tolerance
- Created a insert for the turning 550
- Created a Jig for the Dremel to fix the sliprings

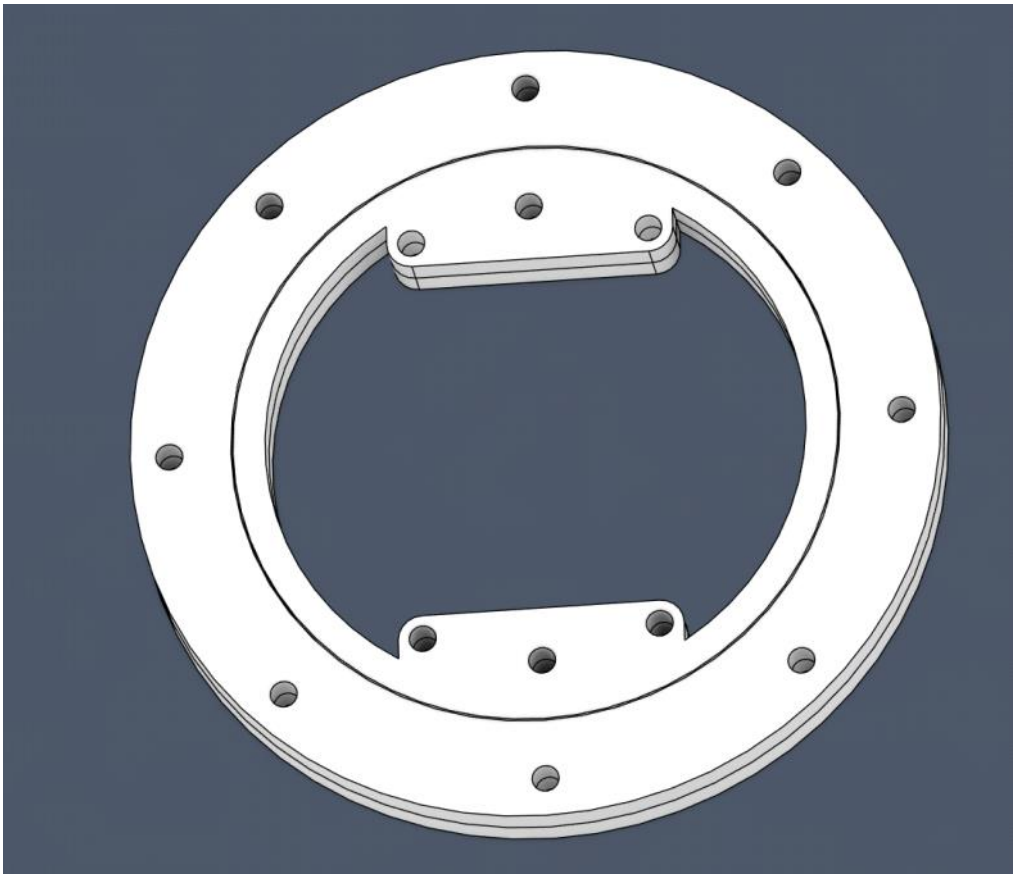


7/6

- ~ 7 hours
- Fixed the Dremel holder L-bracket hole by creating an insert that converted the 7mm to 3.5 mm hole
- Rebuilt CAN sliprings and sanded the inconsistencies out of the sliprings using the Dremel holder
- Modified the brushes and supports to have tension bands on both sides. Having it on one side was created inconsistent pressure issues when going one direction over the other

- Transferred all the wires to the new slipring brushes
- Decided to use Debian Linux on the OrangePi, got VimbaX API working in Anaconda
- Began creating the arm servo joint and bearing and selected the servo
    - Instead of my previous intention of putting all the weight on the servo we decided to added a bearing that would hold the weight for the servo. If all the weight was put on the servo the axle may have broken and the weight may prevent the servo from turning.



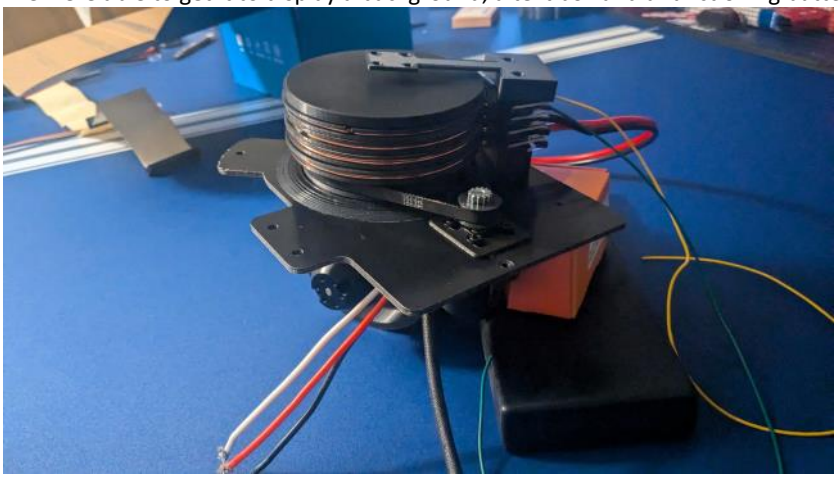


7/7

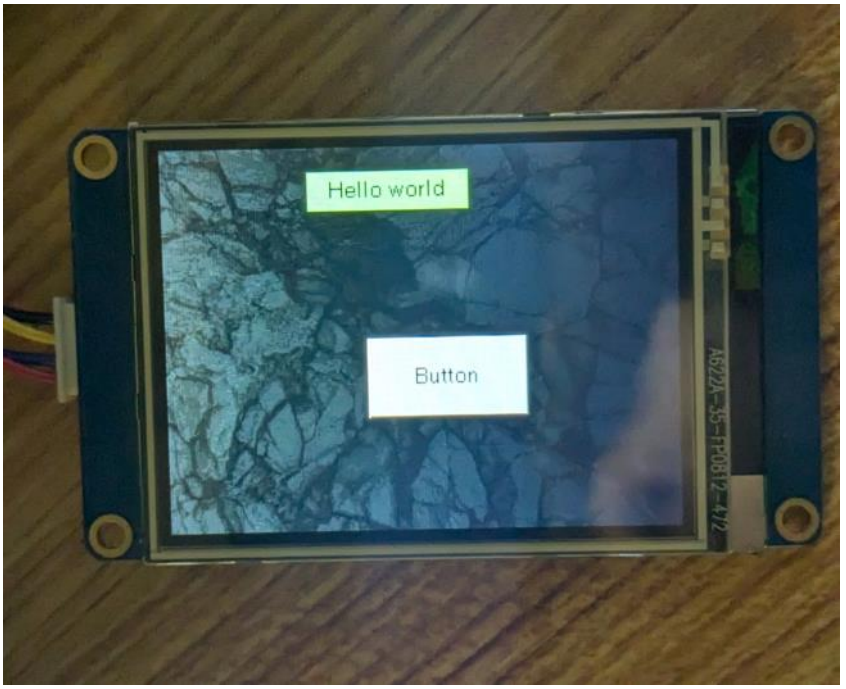
- ~ 7 hours
- Adjusted corner chassis pieces and re-enforced the connecting supports
- Created pin assignments for the Orange Pi and LCD screen
- Remote connected to the Orange Pi via NoMachine
- Worked on debugging Python libraries for the Hosyand LCD TFT with ili9341 chip
    - Had a lot of problems with the spidev library. Turns out that the /dev/spi* permissions have to be manually set
      sudo chmod u+rw /dev/spidev0.0
    - Referenced: GitHub - sonocotta/ili9341-orangepi-python: Python library to control an ILI9341 TFT LCD display on the Orange Pi SBC
- To Do
    - Figure out why GPIO cannot be set
        - Possibly a timeout issue?

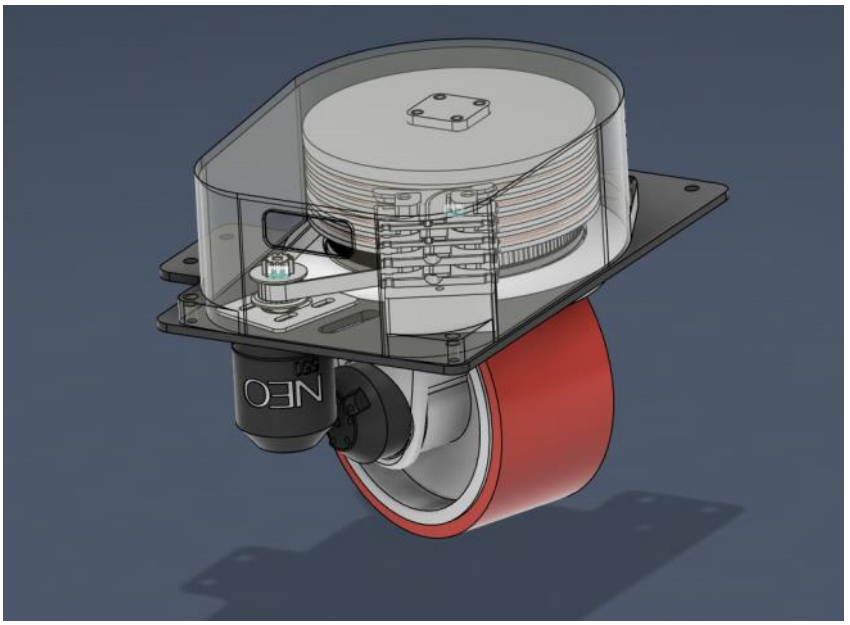
7/8
- ~ 8 hours
- Focused on debugging the LCD screen
    - Used numerous libraries on the OrangePi
    - Spent a lot of time attempting to circumvent the gpio/direction permissions
    - Found an problem where enabling both chip selects on the Spi0 caused an issues where none of the Spi bus would function (pins did not toggle)
    - After fixing the chip select we were finally able to send and receive using the spi. Previously the oscilloscope showed nothi ng on the SCLK, CS and the MISO and MOSI pins were not functioning as normal. After the fix everything was working as expected; however the screen was not displaying anything we sent.
    - We found out that we had been using packages for another LCD screen - ili9341 which was similar to the 9488
    - After a while we moved to an Arduino Esp8266 and attempted to utilize the Arduino_GFX provided by moononournation: GitHub - moononournation/Arduino_GFX: Arduino GFX developing for various color displays and various data bus interfaces
    - Arduino_GFX : 31 Steps (with Pictures) - Instructables
    - Moononournation's Arduino_GFX looked promising; unfortunately, we could not get it to work with our setup
    - After using various libraries and other packages, and externally powering the LCD we found no success.
    - After searching for other solutions a library for the OrangePi created by Adafruit was found
    - We installed Armbian on the Pi instead of Debian and going to see if Adafruit's software can be used

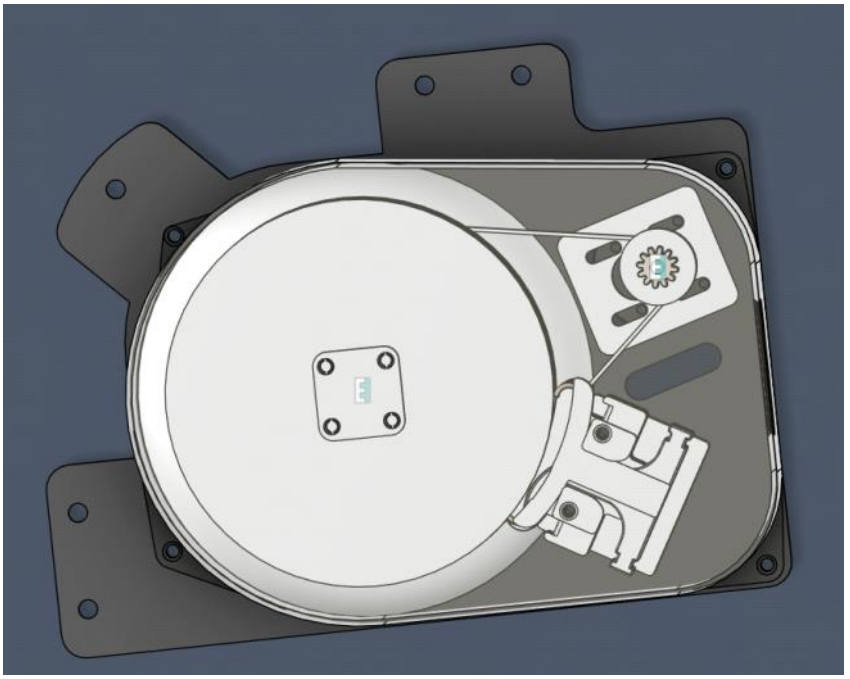
7/9
- ~ 6 hours
- Assembled the second revision of the swerve module
- Began working on a third revision of the swerve module
    - Fixed some tolerance issues in various places
    - Altered the brush stack so they are symmetric and cut the base so that it naturally will lean forward. The intention is that the forward force will counter act the force of the sliprings pushing the brushes backwards.
    - The placement of the 550 and the brush stack were changed so that the space was utilized more efficiently
    - The cover was redesigned to match the new footprint of the module. The mag encoder will now be a part of the cover instead of it being on its own mount connected to the brush stack.
- We decided to move away from the Ili9488 and instead use a Nextion LCD display which uses the UART protocol instead of SPI.
    - We were able to get it to display a background, a text box and a functioning button.



Swerve Module
Rev 2

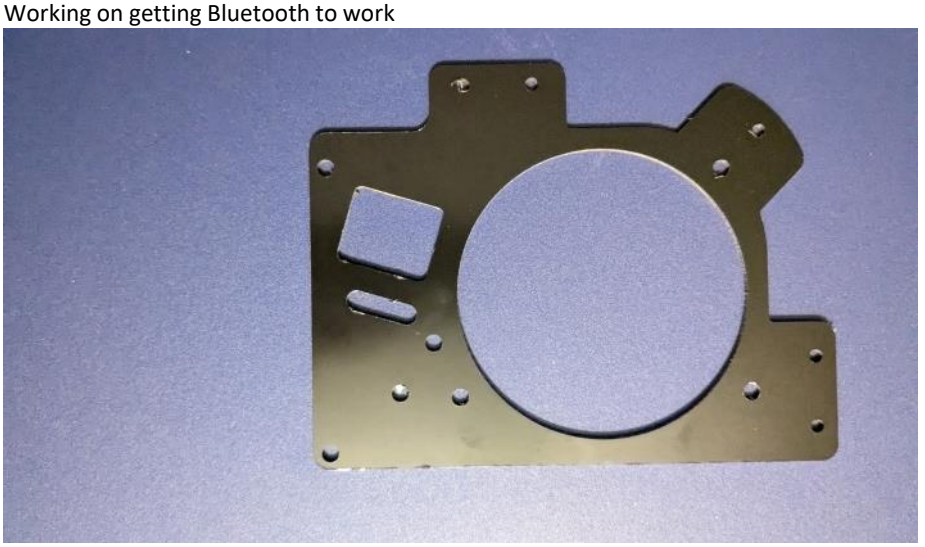

Hello world
Button


NEO



7/10

- ~ 6 hours
- Routered out the third revision of the swerve module plate
- Assembled the third revision of the swerve module and began creating the 4th version
    - Instead of gluing the bearing holder onto the plate we are going to add 6 screws
    - The bearing holder was made taller to accommodate the screws
    - The cover had to be altered so that there are gaps for the new bearing holder
- Found a candidate python library to work with the Nextion screen
- Working on getting Bluetooth to work



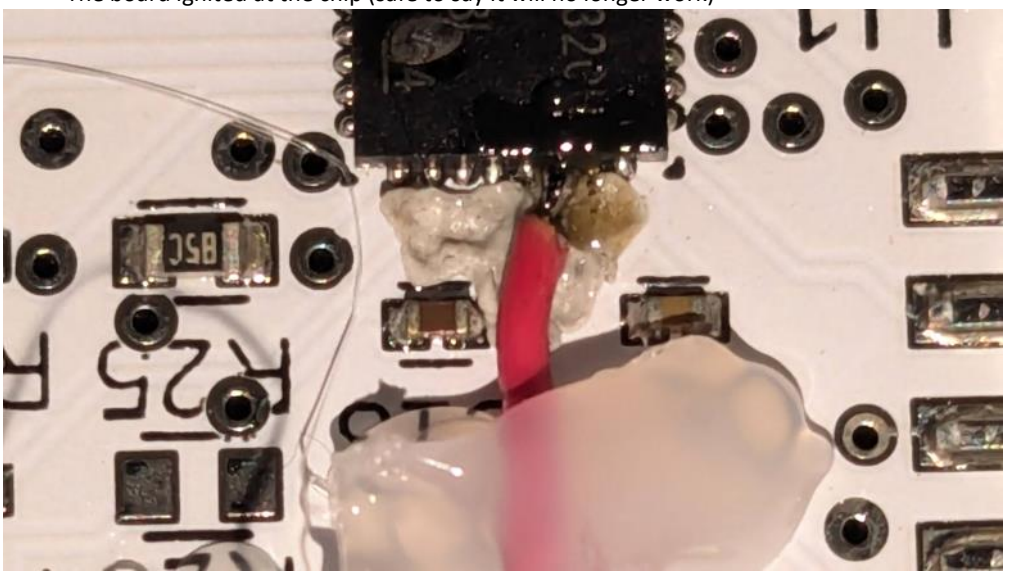
Swerve Module
Rev 3

7/11

- ~ 4 hours
- Working getting a PS4 controller to work with the Orange Pi
- Working on trying to set up another of the motor controller boards to do some more testing with

7/12

- ~ 5 hours
- Working on getting the PS4 controller to work with Arduino
    - Had success once, but we were unable to replicate out success until later
    - We found out that the Esp32 had to be scrubbed with the removed paired devices file in the git hub link in order for the controller to connect
    - Referenced:
        - GitHub - un0038998/PS4Controller_ESP32: This repository contains code and diagram for using PS4Controller with esp32
        - Video: https://www.youtube.com/watch?reload=9&v=dRysvxQfVDw
- Set up another motor controller board and tested it
    - The board ignited at the chip (safe to say it will no longer work)

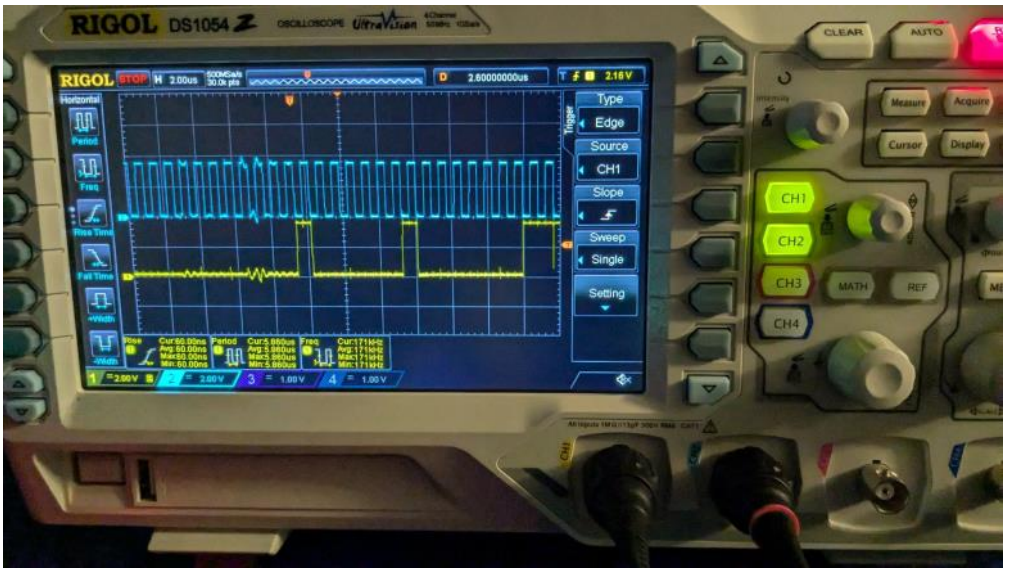


After math of testing

7/13

- ~ 5 hours
- Added cross braces to the chassis corners so they do not fold in when put into the bars
- Got the Orange Pi and the Esp32 to communicate with each other via SPI with: GitHub - hideakitai/ESP32SPISlave: SPI Slave library for ESP32

- Initially, the Esp32 kept sending 0, 1, 2, 3 later we found that the tx_buf was being initialized later and was overwriting t he our data
- Afterwards messages were received and sent both ways. The messages sent from the Orange Pi have some consistency errors that need to be looked into



Esp 32 sending:
0, 1, 2, 3

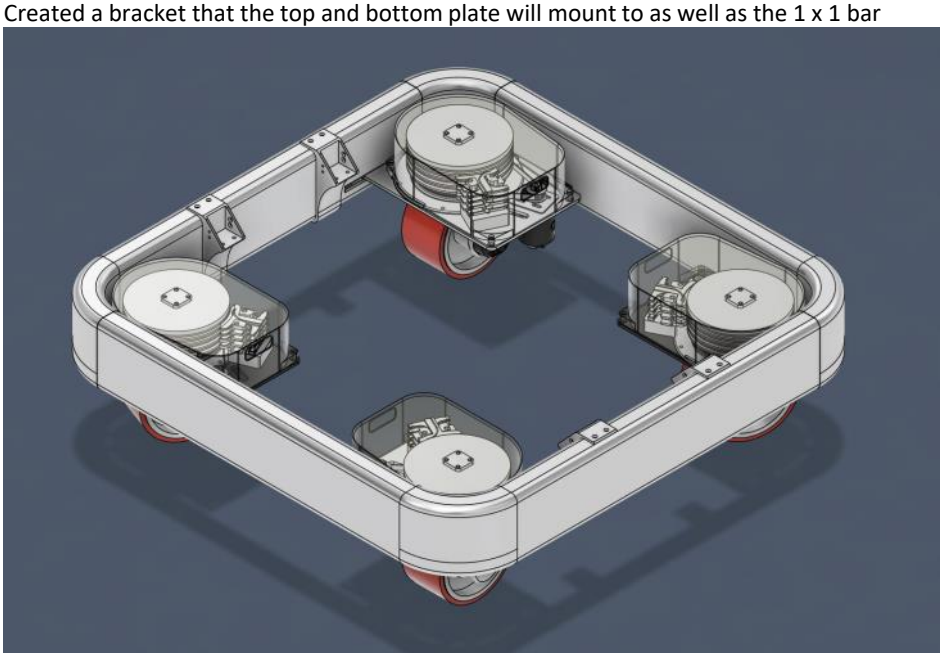
Esp 32 after fixing:
Ca, fe, ba, be

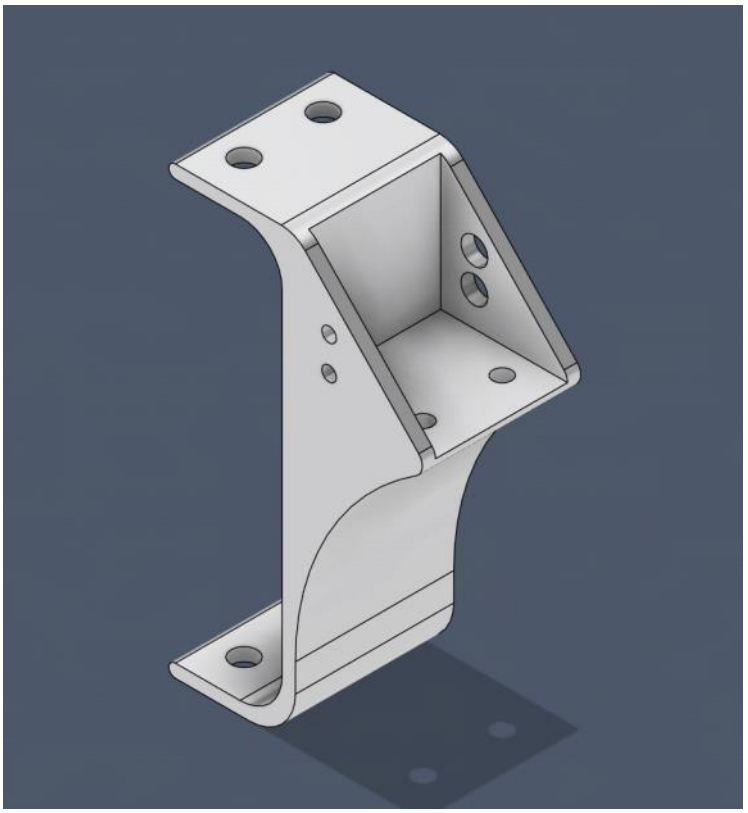
7/14
- ~ 3 hours
- Got the Esp32 to take inputs from the ps4 controller and send them over to the Orange through SPI reliably
  - Have to add 128 on the Arduino side because the SPI library expects a uint_8, but the ps4 function is returning a signed int
  - Once sent to the OrangePi 128 should be removed from the transmitted data
- Introduction · FRC Swerve Drive Programming (gitbooks.io)

7/15
- ~ 1 hour
- Began looking at the motor controller board design and stated making some changes
  - Increased the size of parts to 0805 if they were smaller
  - Pulled the Mosfets together and closer to the DRV8320
  - Decreased the overall package size

7/16

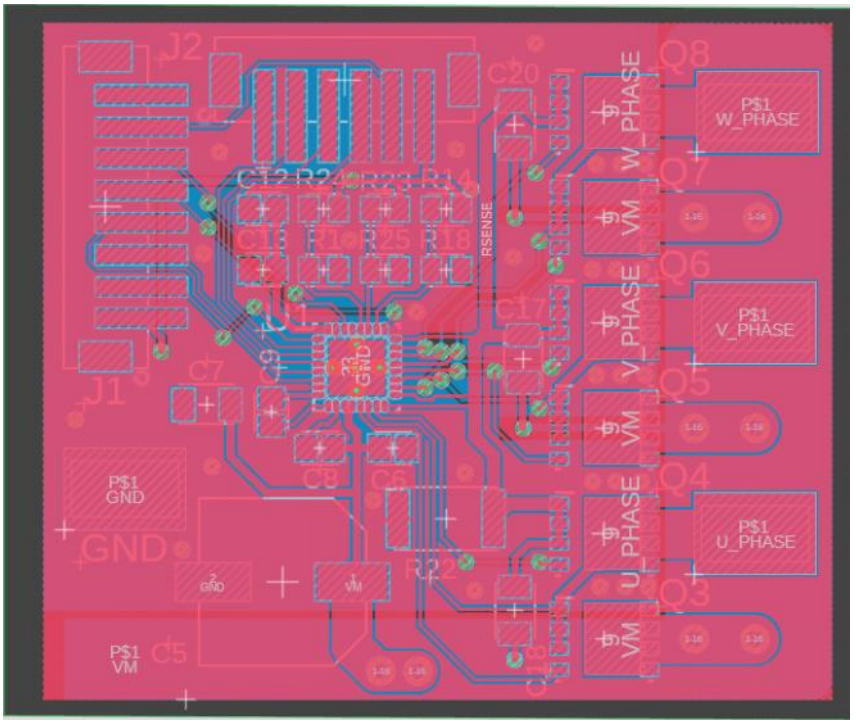
- ~ 3 hours
- Assembled the chassis in CAD
- Created a mirror version of the case and plate of the swerve modules
- Created a bracket that the top and bottom plate will mount to as well as the 1 x 1 bar





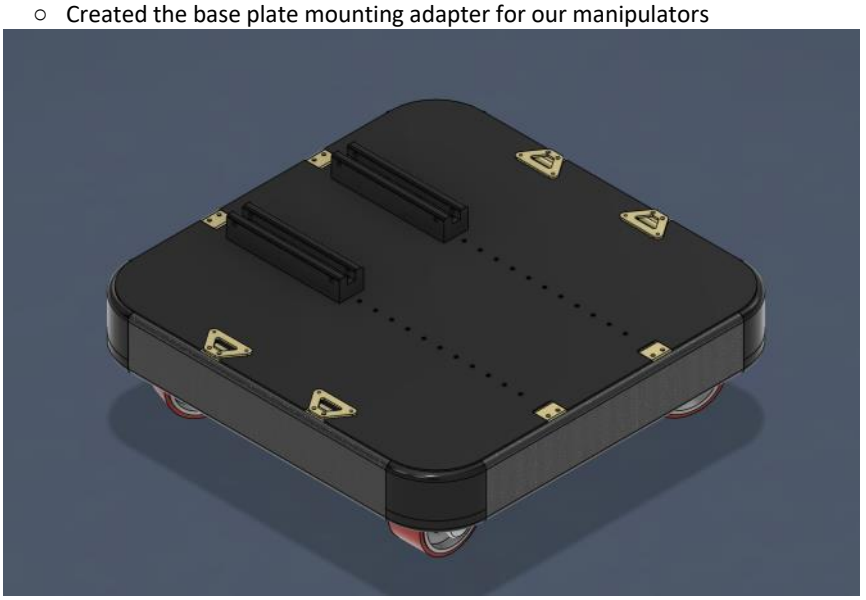
7/17

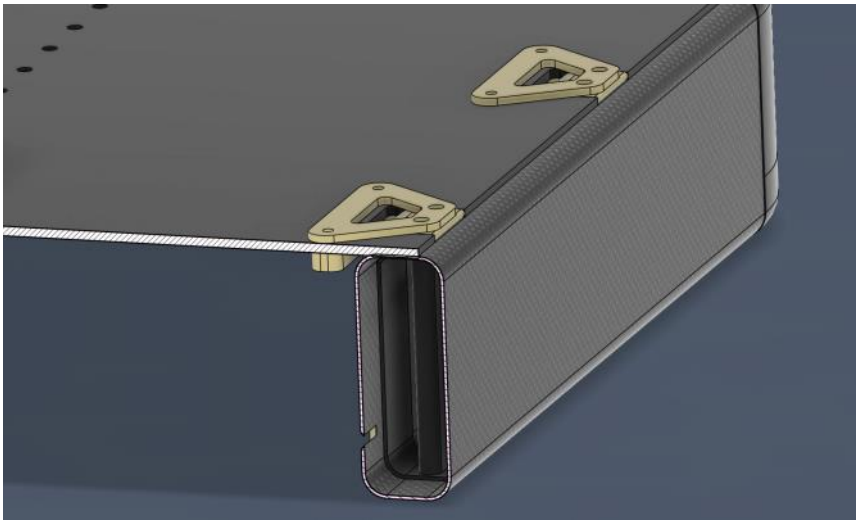
- ~ 5 hours
- Created the Skid plate in CAD and modified the 1 x 1 brackets to attach to the top
    - Looking potentially into using gussets to attach the top plate to the brackets
- Continued board development and finished creating a second revision



7/18

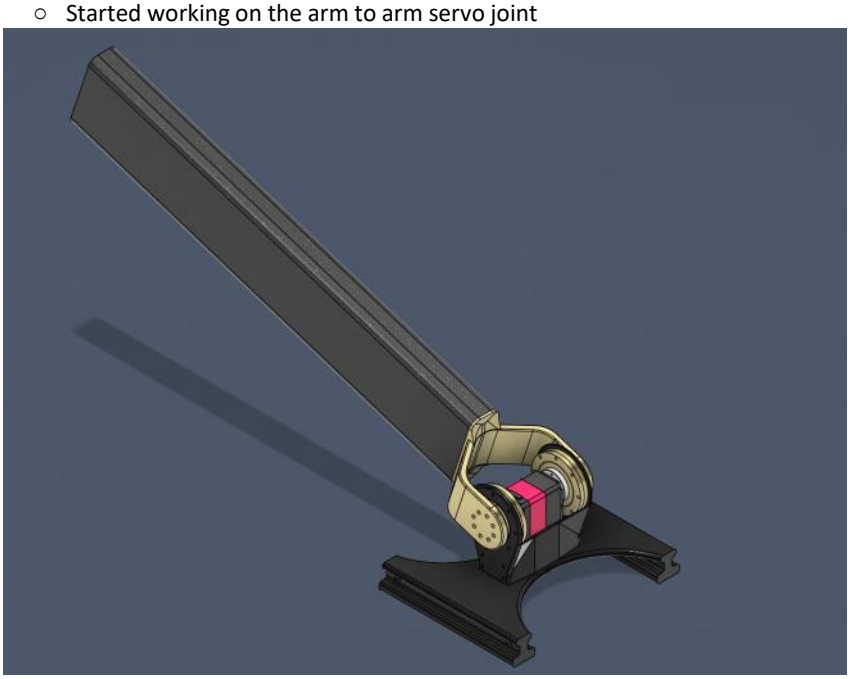
- ~ 4 hours
- Worked on chassis and top plate mounting
    - Decided to design it so there is a mounting gusset that is put onto the base plate and then that is screwed into the chassis
    - Created the base plate mounting adapter for our manipulators





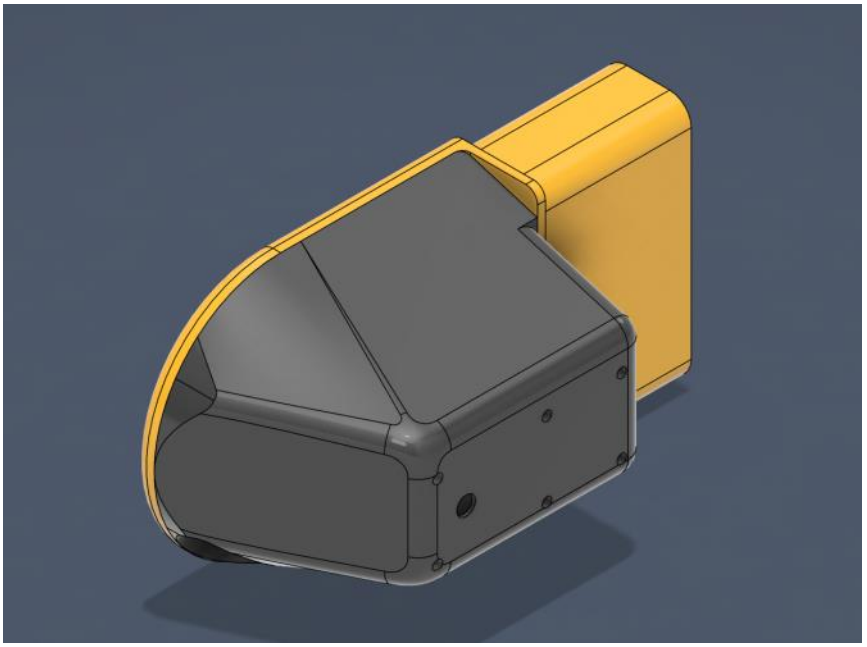
7/19

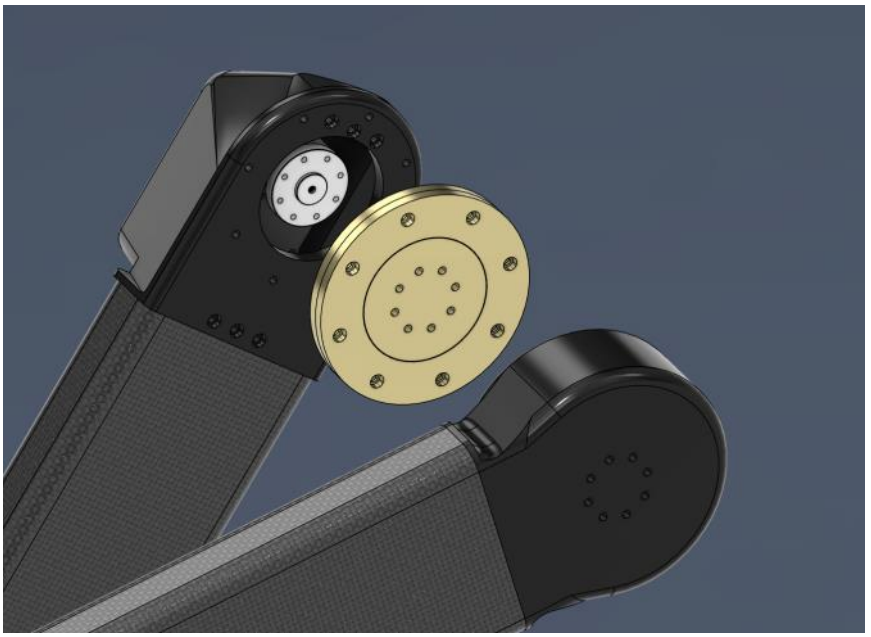
- ~ 6 hour
- Worked on the arm in CAD
    - Created the arm to base mount, the base of the arm, the base bearing, and the arm base to arm bar
    - Started working on the arm to arm servo joint

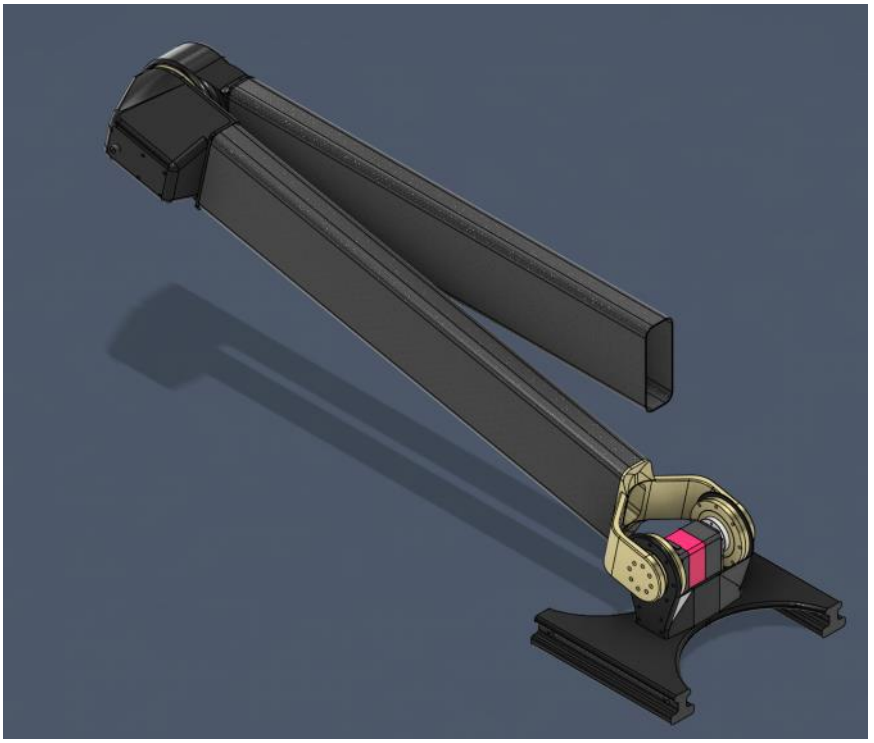


7/20

- ~ 5 hours
- Programming
    - Got Debian onto the Nvme drive, used the Orange Pi 5 user guide and Orange Pi 5 NVMe/SATA SSD Boot Guide - James A. Chambers (jamesachambers.com)
    - Applied all known updates to the OS
    - Tested the Esp32 PS4 connection with the Pi
    - Created a Git repository for the code
    - Started a skeleton of the code
    - Plan to use multiprocessing: https://www.datacamp.com/tutorial/python-multiprocessing-tutorial
- Design
    - Worked on the arm joints
    - Designed a 4 part joint to connect the two arms together

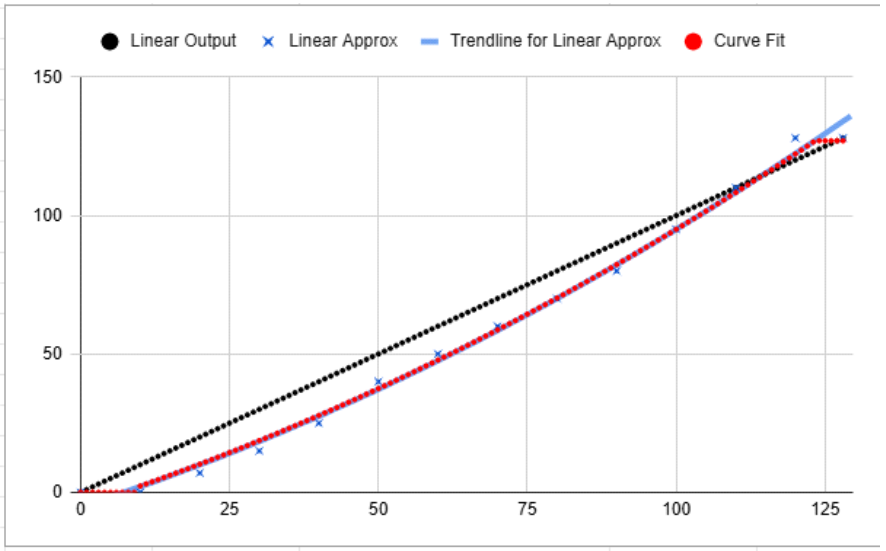






7/21

- ~ 7 hours
- Got the Git repository to work
- Moving from Arduino to VS Code
- Worked on creating a framework for the rest of the code
    - Created set up information that is sent to the ESP32 from the Pi - this includes response curves and error checking on the SPI Bus



Spi Communication		
byte0	0	0 = poly, 1 = sigmoid
byte1	5	constant value, negated by ESP32
byte2	7	x-coefficent, /10 by ESP32
byte3	3	x^2-coefficent, /1000 by ESP32

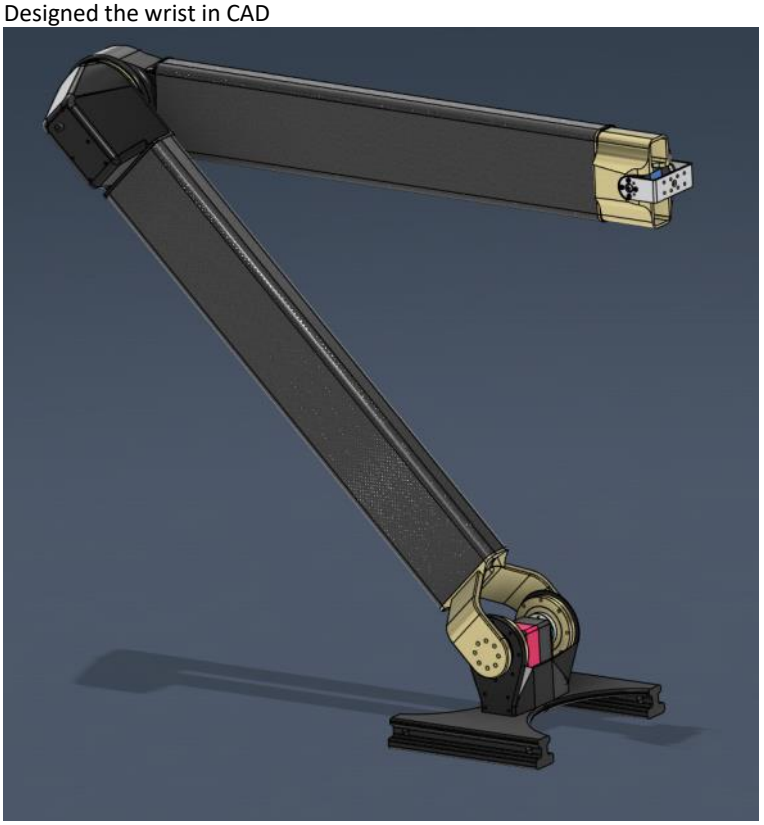
7/22

- ~ 5 hours
- Worked on getting the programming to work

- Had scope issues and linking errors

7/23
- ~ 2 hours
- Attempted to fix the linking errors

7/24
- ~ 1.5 hours
- Designed the wrist in CAD



7/25
- ~ 3 hours
- Finally resolved the issue of the ESP32 sending 0s on SPI: the tx buffer size needs to be a multiple of 8 bytes
- Fixed the curve coefficient communication handshake
- Implemented the response curve on the ESP32 (complies, not yet tested)
- Started a "Pi Startup" tab in the spreadsheet to describe the Orange Pi interaction with the LCD, ESP32, FPGA, Swerve, and Arm

7/26
- ~ 2 hours
- Started writing the PWM spec document

7/27
- ~ 4 hours
- Completed the first revision of the PWM spec
- Installed and debugged the VS code toolchain for the Tang Nano using: Lushay Labs
- Wrote RTL for the PWM and wrote a test bench, and confirmed all the basic operations of the PWM are working as expected

7/28
- ~ 3 hours
- Wrote the first revision of the SPI spec

- Wrote RTL for the SPI and wrote a test bench, and confirmed all the basic operations of the SPI is working as expected

7/29
- ~ 6 hours
- Wrote the first revision of the Address Decoder spec
- Started FPGA Sub-system spec
- Wrote RTL for the Address Decoder
- Identified some updates that need to be applied to the PWM

7/30
- ~ 5 hours
- Created test bench for Address Decoder
- Started working on PWM Control and PID spec
- Started thinking about how to implement PID tuning and I2C into the system

7/31
- ~ 3 hours
- Started working on PWM Rotation Controller spec

8/1
- ~ 2 hours
- Wrote RTL for PWM Rotation Controller

8/2
- ~ 2.5 hours
- Finished writing RTL for PWM Rotation Controller and wrote test bench

8/3
- ~ 3 hours
- Started writing RTL for pwm_ctrl

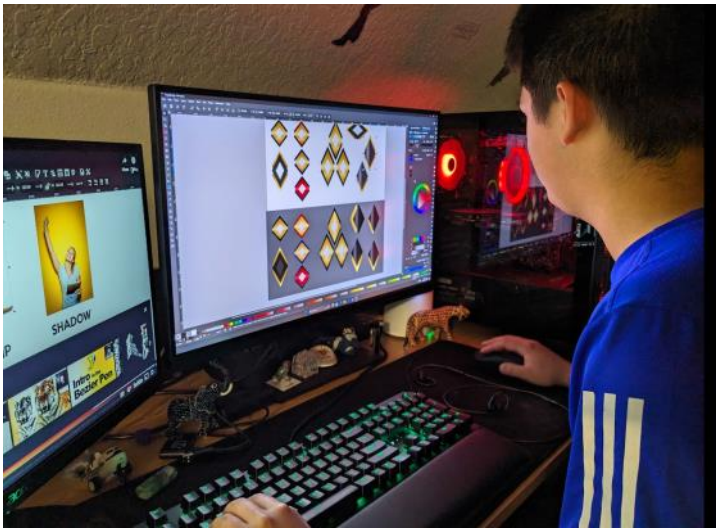
8/5
- ~ 5 hours
- Designed the first functional grabber for the robot
- Finished writing RTL for pwm_ctrl, and modified Lushay Labs' I2C code (preventing the D latches from being formed; and reformatting to our preference)
- Created the top level file and got the RTL code to build

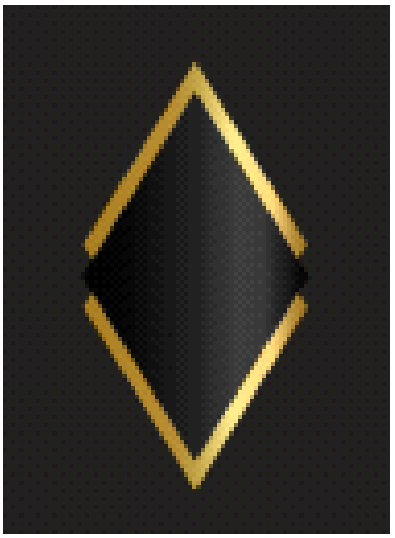
8/6
- ~ 3 hours
- Began working on a new version of the grabber

8/7
- ~ 3 hours
- Finished new version of grabber and updated the arm lengths

8/9

- ~ 6 hours
- Worked on creating a logo





8/10

- ~ 4 hours
- Worked on animating logo and began ordering more supplies for the robot

8/11

- ~ 4 hours
- Finished animating logo with Da Vinci Resolve and began 3D printing a tire

8/12

- ~ 2 hours
- Worked on creating the carbon fiber base plate
-

- 

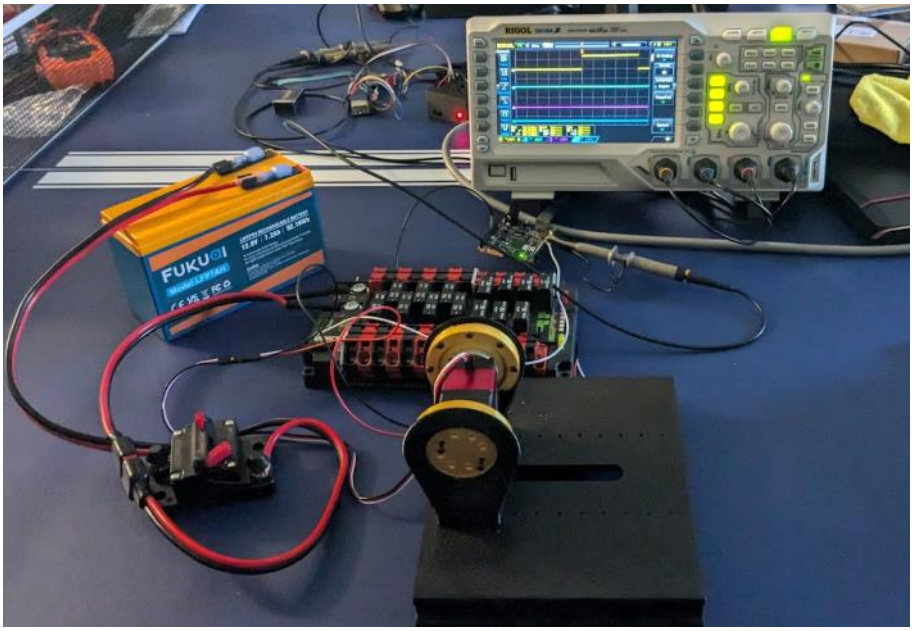
8/13
- ~ 2 hours
- Worked on the website

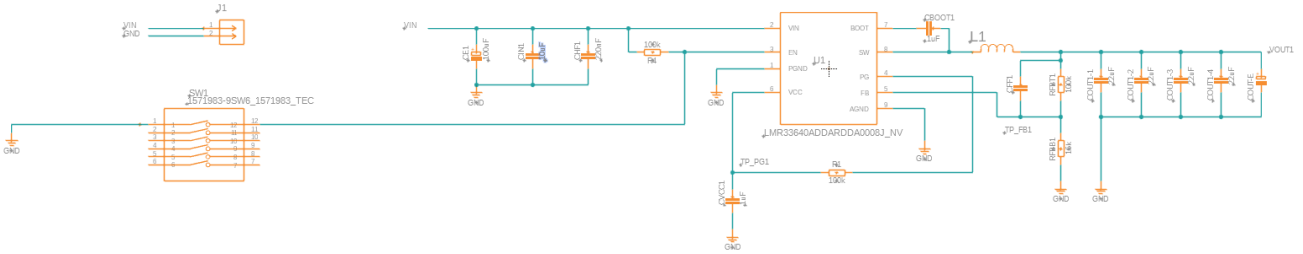
8/14
- ~ 3 hours
- Worked on the website and pushed the first revision to Github

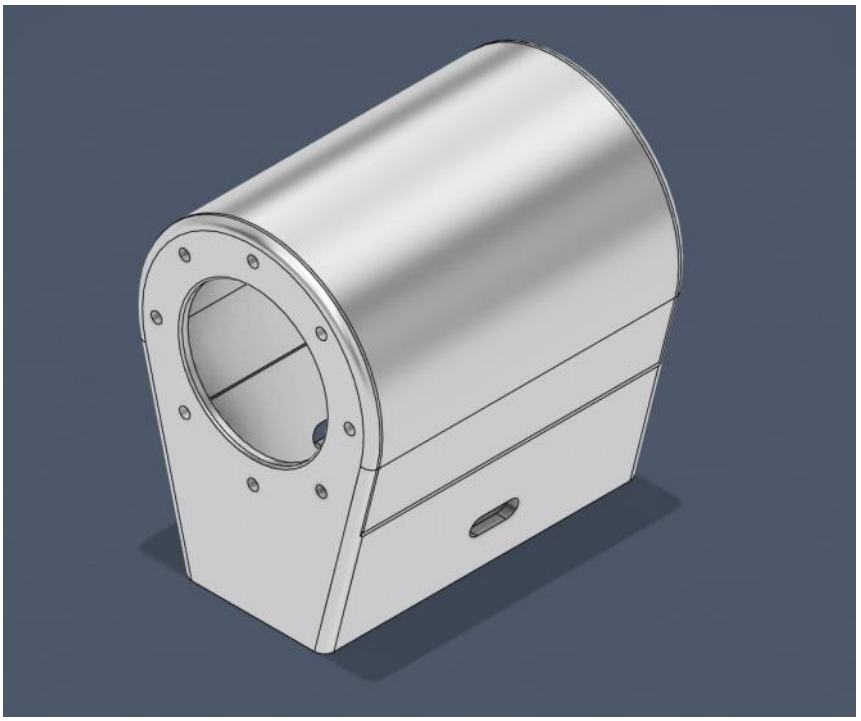
8/15
- ~ 3 hours
- Worked on website
- Started 3D printing and building arm bearings
- Started looking at the voltage regulators for the motherboard

8/16
- ~ 2 hours
- Worked on getting parts 3D printed

8/17
- ~ 5 hours
- Tested gold paint of the gold 3D printed parts
- Tested the 150kg PWM servo to make sure it works
- Redesigned base mount to make it more robust
- Worked on the motherboard and created the schematic for the first voltage regulator (12V)

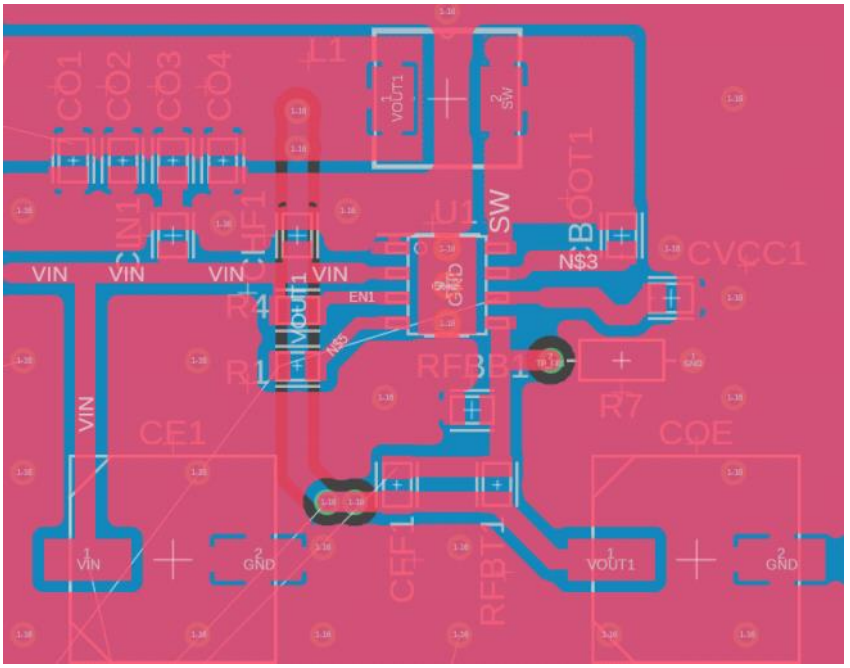


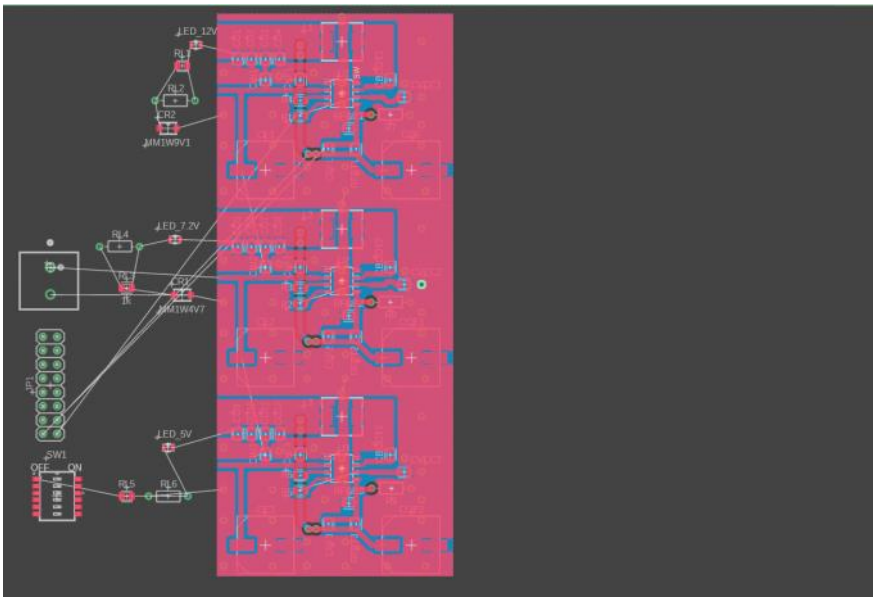




8/18

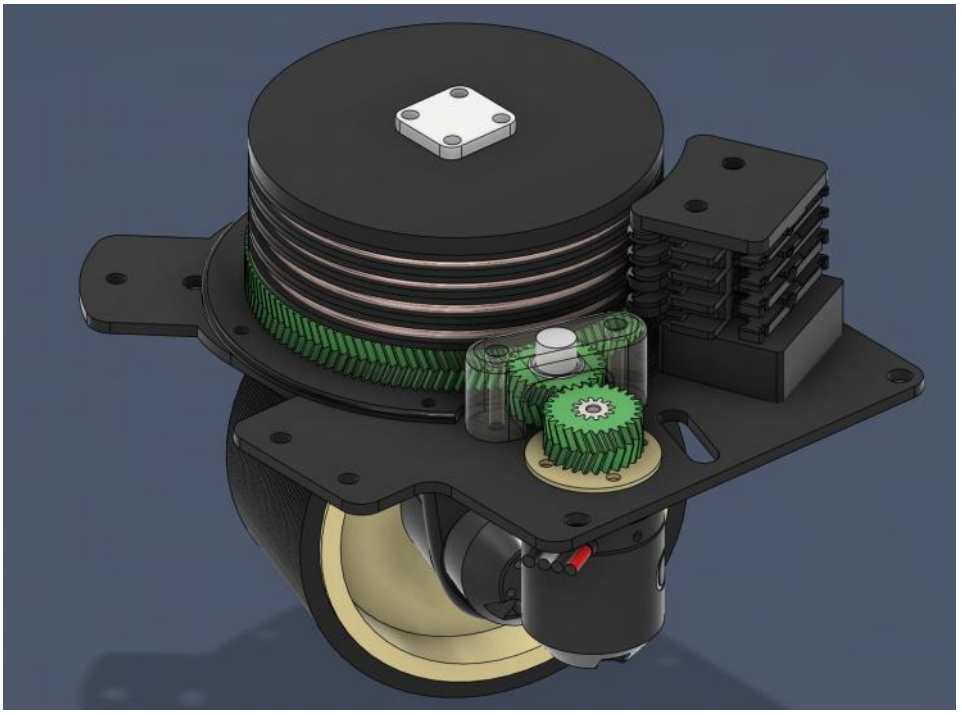
- ~ 4 hours
- Worked on motherboard
    - Finished creating each of the adjustable voltage regulators for 12V, 7.2V, and 5V

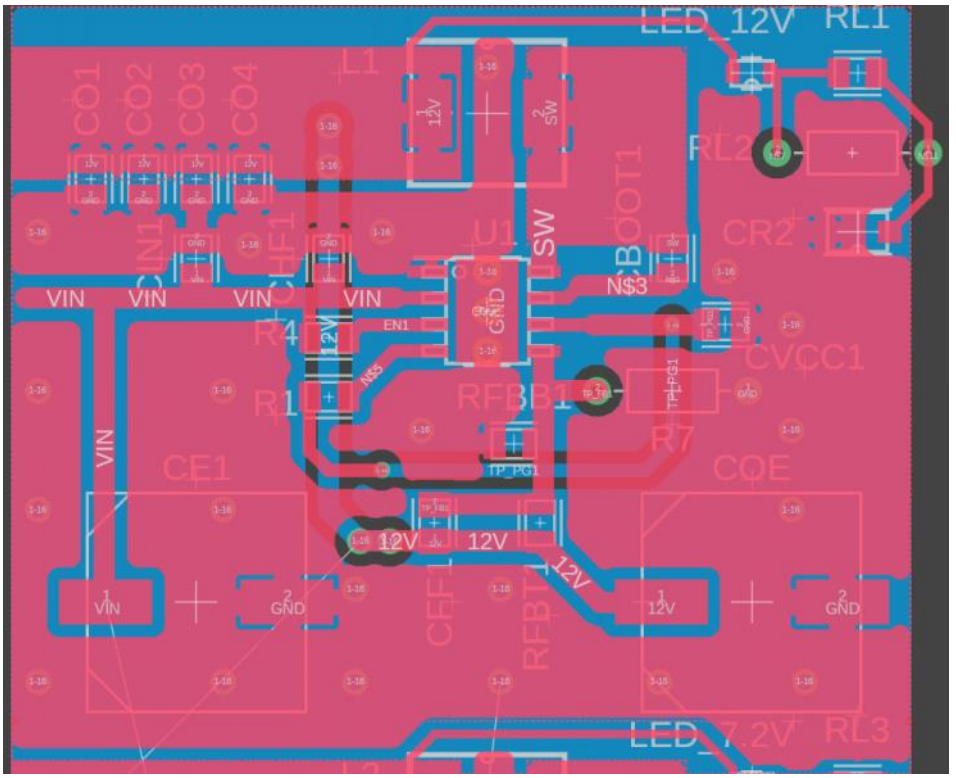




8/19

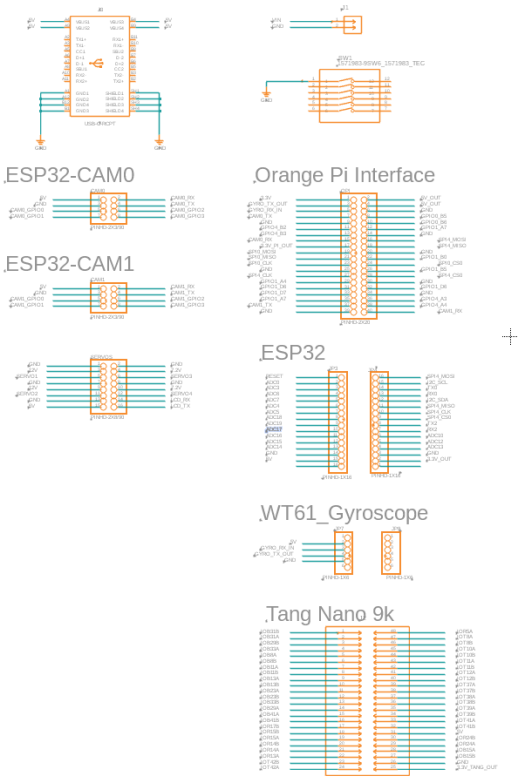
- ~ 5 hours
- Converted the rotation on the swerve modules from belts to gears
- Worked on the motherboard
    - Connected the LEDs and added the 40 pin header for the Orange Pi
    - We are thinking about what to separate from the motor feedback to the Orange Pi and the Tang Nano 9k
- Started to print the other 3 swerve modules

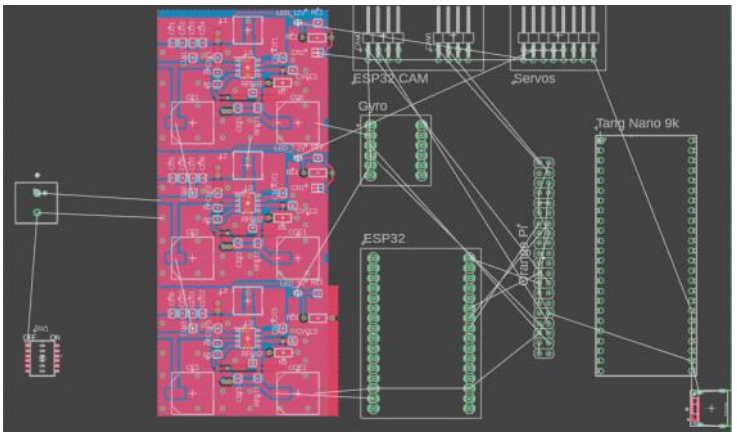




8/20

- ~ 6 hours
- Worked on the mother board design
    - Added the Orange Pi, ESP32, ESP32 Cams, and Tang Nano interfaces
- Worked on building the arm, and continued to manufacture parts for the other swerve modules





8/21

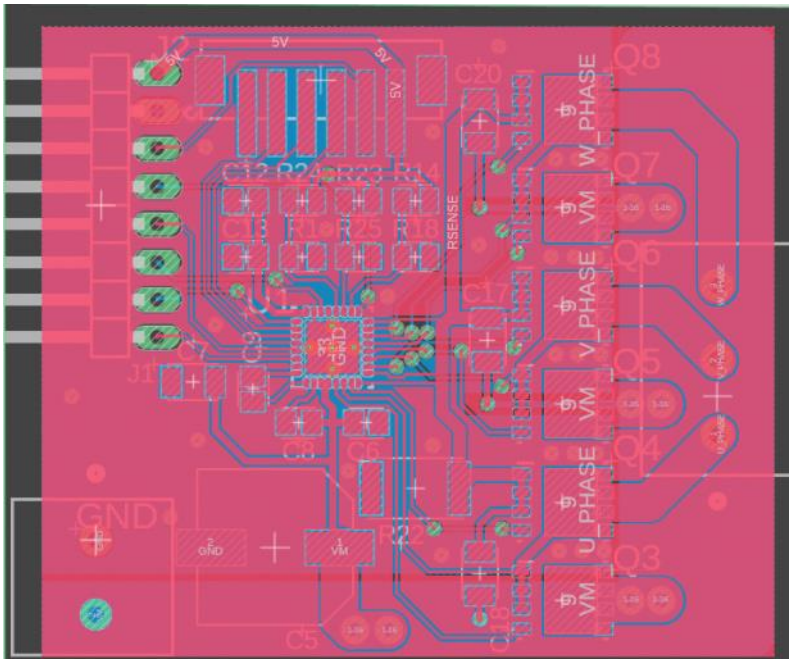
- ~ 2 hours
- Looked at board design
    - Looking at connector for the power wires to the boards

8/22

- ~ 3 hours
- Worked on getting the carbon fiber weave onto the fiberglass bars

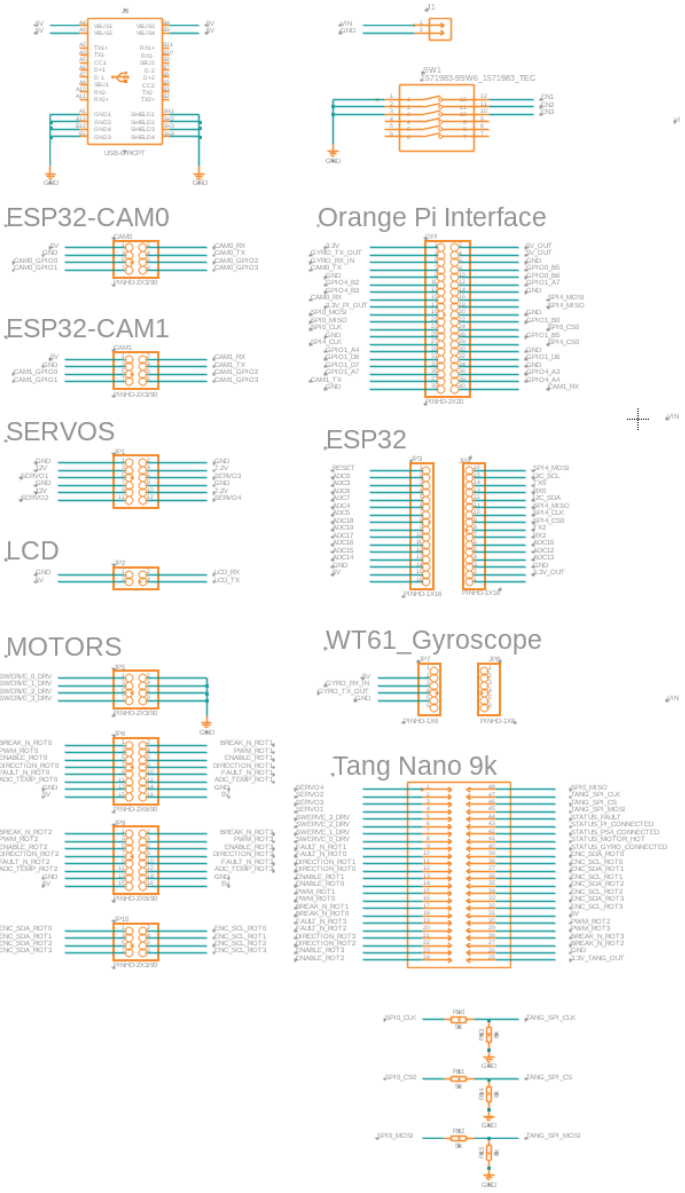
8/23

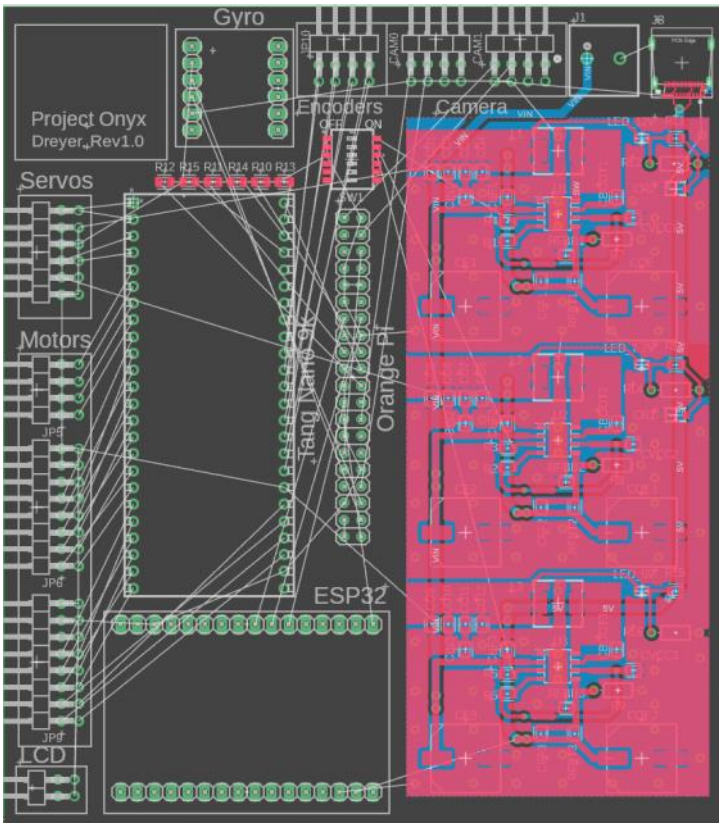
- ~ 2 hours
- Worked on DRV8320 to replace the connectors



8/24

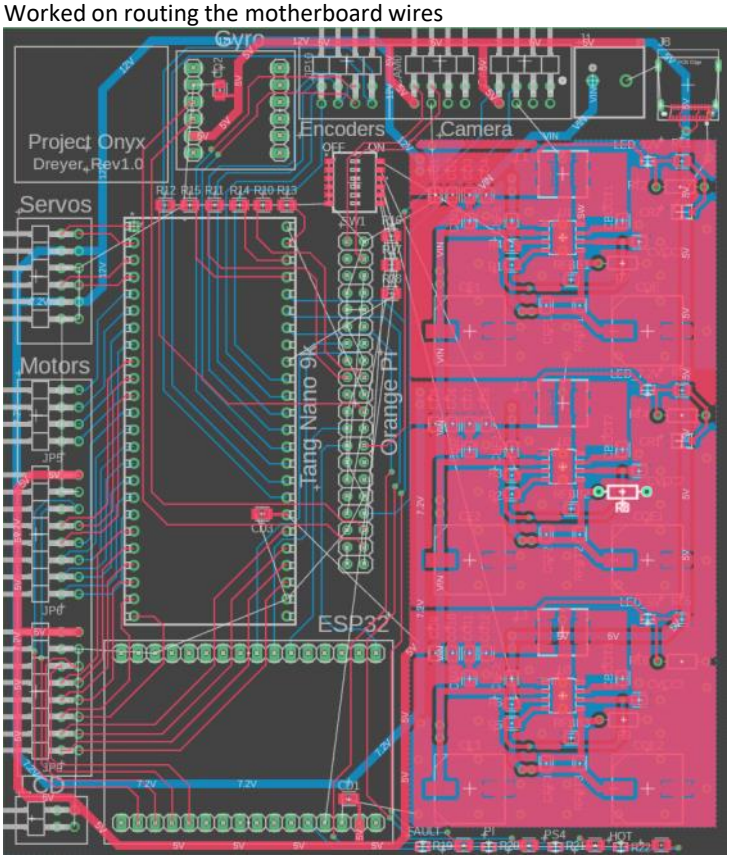
- ~ 4 hours
- Worked on the Motherboard
    - Added missing connectors and assigned all the Tang Nano 9k pins
    - Reoptimized the position of all the parts


ESP32-CAM0
Orange Pi Interface
ESP32-CAM1
SERVOS
ESP32
LCD
MOTORS
WT61_Gyroscope
Tang Nano 9k



8/25

- ~ 4 hours
- Worked on routing the motherboard wires



8/26

- ~ 3 hours
- Finished routing the Motherboard