Development Board Tester

Design Document

Prepared By:
Conner Inglat
Dylan Vetter
Corey Moura
Nathan Hanchey

August 5, 2022
Sponsor: GVSU

Design Document Review Sign-off Sheet

Dr. Karl Brakora – GVSU Sponsor Contact

Signature _____ Date _____

Dr. Nicholas Baine – GVSU Team Advisor

Signature _____ Date _____

Corey Moura– Team Captain

Signature _____ Date _____

Nathan Hanchey– Sponsor and GVSU Advisor Point of Contact

Signature _____ Date _____

Connor Inglat– Treasurer

Signature _____ Date _____

Dylan Vetter – Secretary

Signature _____ Date _____

Executive Summary

There are several engineering courses where students are required to program and build hardware circuits with their microcontroller. Hardware connections made incorrectly can destroy or damage individual pins and in some cases the entire board. Additionally, some boards are damaged before being used by the student.

Students and professors often have the need to identify whether unusual behavior is hardware or software related in embedded system projects. Oftentimes, it can be quite cumbersome to diagnose hardware related issues that may occur on a microcontroller, such as damaged GPIO pins, voltage regulators, and other critical hardware peripherals. This presents a need to find a more expedient way to diagnose these issues when many students may require help.

This project will solve this problem by checking the functionality of the Arduino Uno and STM microcontroller hardware through a series of hardware tests. This will allow students and professors to confirm hardware faults or rule out any hardware errors thought to be causing issues. If hardware is not behaving properly, the device created by this project will display the hardware failures on a Graphical User Interface. Otherwise, the student or professor can assume their issues to be coming from software.

Table of Contents

Summary of Project

The MicroDev device is a hardware testing fixture for the Arduino Uno and the STM32 Nucleo F446 development boards. It's designed to confirm that the microcontroller, or "subject board", has functional analog and digital pins. If there is a malfunctioning part of the board this fixture can identify where the problem lies. This project is broken down into three main categories: enclosure, hardware and PCB design, and software.

The curriculum at GVSU uses the Arduino Uno and STM32 Nucleo in foundational Engineering courses. While developing basic skills with embedded programming it is important to know the functionality of the board. For these earlier courses, the professor's time can be taken up by students unable to identify whether the microcontroller is damaged or their program has a fault. This testing fixture is meant to solve that problem in a convenient and quick way. The software design also has modular aspects to make it easier to upgrade with curriculum changes and alternative subject boards.

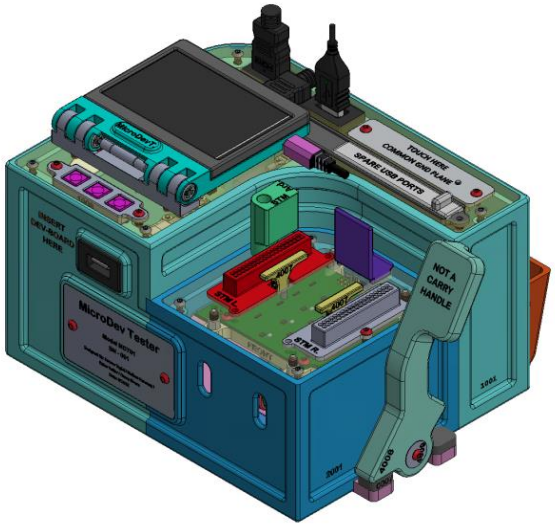


Figure 1.1: Isometric view of device

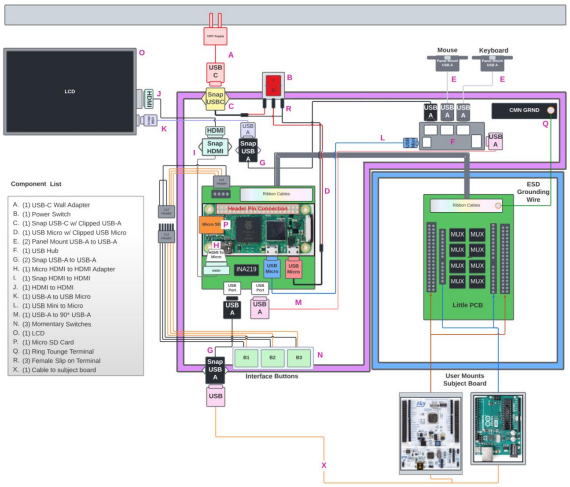


Figure 1.2: Top view of enclosure showing cable routing

Specifications Update

The following specification has been removed from the original sign-off document. Current from the 5V and 3.3V pin will not be measured, but current draw of the subject board will be measured. No additional specifications have been added or removed. A full list of specifications is found in **Appendix C**, and the original scoping document can be found in **Appendix N**.

Table 1. Altered Specifications

Req #	Requirement	Spec. #	Specification
Removed			
3.01	The device must test and record the **current supplied** from the subject board's **5V and 3.3V output pins**	3.01.1	The device must test that current remains below a configurable threshold
3.08	The device must test and record all of the		

	pins capable of GPIO wakeups from sleep mode		
5.00	The STM device production quantity must be 5	5.00.1	There must be 5 total testing devices produced
6.00	The Arduino device production quantity must be 5	5.00.1	There must be 5 total testing devices produced
Added			
5.00	The fixture production quantity must be 10 fixtures.		There must be 10 total testing devices

Budget

This project has been approved for $2,798.00. The cost breakdown by category can be seen in **Figure 1.3**. The projected remainders are calculated based on a subtracted allowance of $420. This allowance will account for tax, shipping costs, and any unforeseen costs that may arise. The quoted values for cable costs and specific hardware components may decrease if they are purchased through the university. In the case that the budget has expired, the sponsor has agreed to delay the purchase of the Raspberry Pi Zero 2W until supply shortages have been resolved, returning their price (from $54-$125) back down to MSRP $15.

Budget: $2,798.00
Allowence: $420.00
Production Qty: 10
Budget Per Tester: $237.80

Orders to be placed with Hector Garcia

Unit Total: $261.44
Remaning: -$23.64

Grand Total: $2,679.34
Remaning: $118.66

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
CNC BOM											
								CutTime		Hrs	
3D PRINT BOM											
								Print Time Total:	31:16:00	Hrs	312.5
RASPBERRY PI / LCD BOM											
								Unit Total:	$89.00	Total:	$890.00
PCB BOM											
								Unit Total:	$56.43	Total:	$564.32
CABLES											
								Unit Total:	$71.53	Total:	$715.25
SWITCHES											
								Unit Total:	$2.77	Total:	$27.70
HARDWARE / FASTENERS											
								Unit Total:	$41.71	Total:	$482.07
Additional Costs - Estimated											
										Total:	$280.76

Figure 1.3: Budget Breakdown by Category

During the later developmental cycle of this project, the original $2,798.00 budget was expanded to $3,048.00 to account for additional costs. These costs came from design revisions and supply chain challenges. The total amount spent on the project is depicted in **Figure 1.4**.

Figure 1.4: Total Budget Utilized

This led to a revised BOM as well. In the revised BOM, the cost of producing 10 complete MicroDev Testers came out to $2,896.89; likewise, the cost to produce a single MicroDev Tester ended up coming out to $289.69. It should be noted that these calculated values assume that both the LCDs and Raspberry Pi Zero 2Ws are priced at a fair market value and not the inflated pricing seen at the time this project was created. Additionally, it also assumes that some parts can be provided through GVSU's lab supplies. The revised BOM can be seen below in **Figure 1.5** and an itemized list of the budget can be found in **Appendix A**.

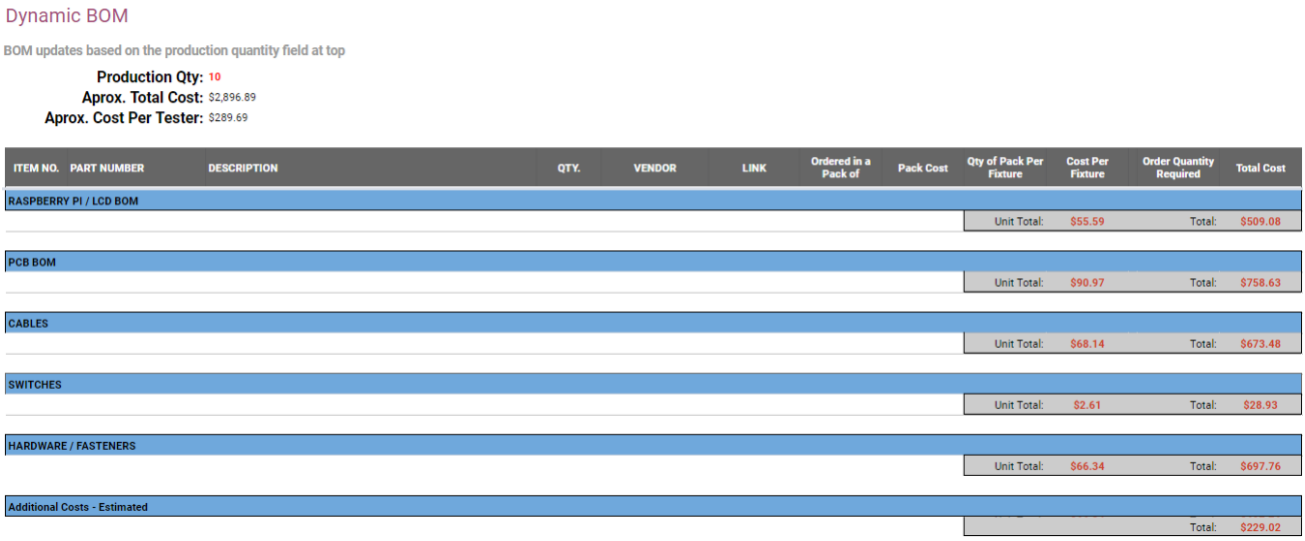


Figure 1.5: Revised Budget Breakdown by Category

Test Specifications and Results

No physical prototypes have yet been built. The devices and software used to validate our concepts are outlined in **Table 1**.

Table 2. Design Sections

Section	Verification Apperatus	Result
Mechanical	Solidworks, digital calipers, physical components to verify model sizes and clearances	Proper theoretical clearances and tolerances for mating components and assemblies
Hardware	DMM, Power Supply, Oscilloscope, and LTspice simulations	LTSpice simulations (**Appendix E**) are compared to hand calculations through analysis of the simulated results. Lab equipment will be used to verify the circuit's real-world functionality through empirical testing.
Software	Raspberry Pi OS tools, Arduino Documentation, STM documentation	Implement a viable software design

Proof of concept testing is outlined in the **Design Sections** of this document. Each section contains their respective validation methods.

Concept Selection

During the initial brainstorming phase of the project, several concepts were generated for consideration. Concepts were generated for the circuit design, enclosure design, and software GUI design. For the circuit design, it was decided that a custom PCB would be better than using prebuilt breakout boards. For the enclosure design, a cheaper and simpler design was chosen over more complex designs, while maintaining practical capability. For the software GUI interface, it was decided to use external push buttons instead of a touchscreen interface. These alternative concepts can be seen in **Appendix M** in more detail.

Enclosure Design

The fixture design uses various plastics in its construction: HPDE, polycarbonate, carbon fiber nylon, fiberglass and PLA. The top plates of each of the two enclosures are CNC milled from clear polycarbonate while the large bottom plate and the PCB mounting plate will be CNC milled from HPDE. The remaining components will be 3D printed using standard levels of infill with a resolution of 0.1mm to 0.3mm. The 3D printed tolerances for loose fit is 0.3mm, for slip fit components is 0.2 mm, and the tolerance for press fit components is 0.1 mm (based on 3D printer specifications). All fasteners are M3x0.5 and have varying lengths. All heat inserts are threaded for M3x0.5 bolts and are 4mm in length. All component drawings can be seen in **Appendix B**.

Enclosure Design Specifications

Table 2. highlights the requirements and specifications that influenced the enclosure design choices shown in the following sections.

Table 4. Enclosure Design Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
1.00	The device **must fit on a desktop**	1.00.1	The device must have overall dimensions not exceeding 1' x 1' x 1'
1.01	The device must be **portable**	1.01.1	The device must weigh less than 10 lbs
		1.01.2	The device must not need additional calibration when moved
1.02	The device must be physically **assembled into a single piece**	1.02.1	The device must not have unattached or unsecured components
1.03	The device must be **simple to load and unload without causing damage to the subject board**	1.03.1	The device must use an alignment post to aid in loading and prevent damage
		1.03.2	The device must use a mechanism to easily and safely remove the subject board without causing damage
1.04	The device **must have an enclosure that can withstand the rigors of daily use**	1.04.1	The device must have 3D printed parts with an infill of at least 20%
2.00	The device must utilize a physical **switch to control main power**	2.00.1	The device must use a toggle switch to control the main power supply
2.04	The device must have a **servicable hardware interface** to the subject board	2.04.1	Must have a header that connects to the I/O pins of the subject board

1.0 Enclosure Overview

The top assembly is composed of two enclosures that slide together via four dovetail joints which share a common base plate for structural rigidity. The larger enclosure will house the majority of the fixture's electrical components while the smaller enclosure houses the subject board's tool pack assembly, and the ejection crank assembly.

Note: The enclosure housings are combined into a single component for 3D printing purposes / manufacturing.

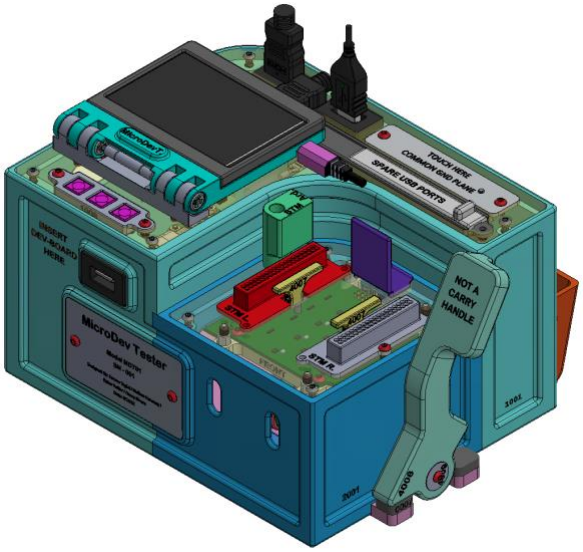


Figure 1.6: Top Level Assembly

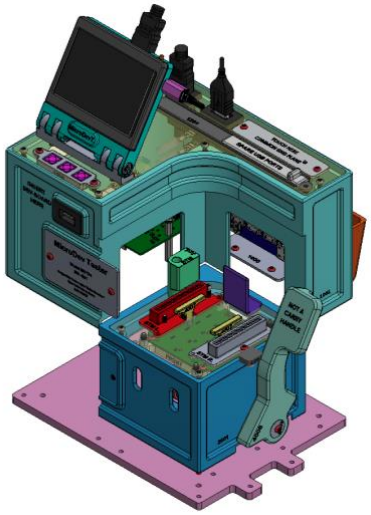


Figure 1.7: Top Level Exploded View

2.0 Large Enclosure Assembly

The large enclosure houses three tactile switches, three USB-A ports (one in front, two hidden under access panel), a four-port USB-A hub, a single HDMI port, a single USB-C port for main power, and a toggle switch for main power shutoff. It will also house the Raspberry Pi Zero 2W, and an LCD.

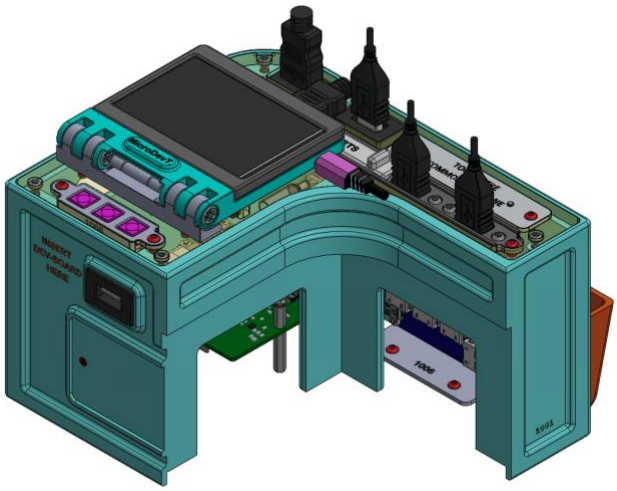


Figure 2.1: Large Enclosure Assembly

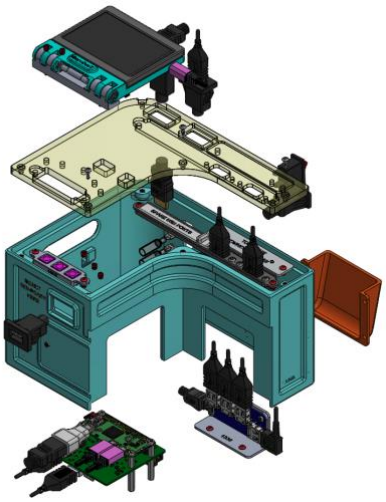


Figure 2.2: Exploded View - Large Enclosure

2.1 Subject Board USB-A Port

A second cutout is located on the front of the large enclosure and is used to connect the subject board's USB-A plug. This connection will provide power to the subject board and also relay serial data for testing.

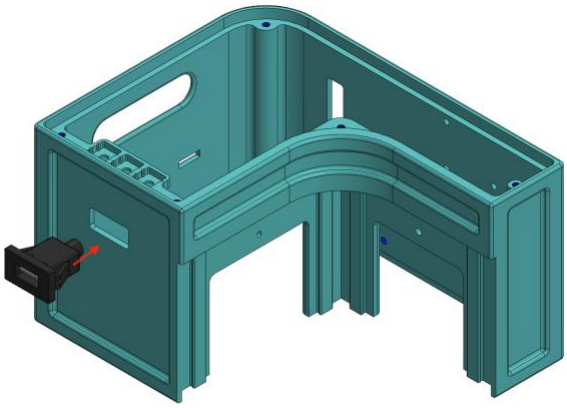


Figure 2.5: USB-A Port Cutout

Figure 2.6: Snap-in USB-A Port Installed

2.2 Tactile Switch Platform and Cutout

A small platform is located on the inside of the front wall of the enclosure. This platform feature has multiple holes which allows the electrical wiring from the three momentary switches to be inserted through, into the space of the enclosure. The momentary switches are secured to the platform with double sided tape and from above via the button cover. The buttons sit slightly above the surface. This design choice helps protect the switches and increase their longevity.

Figure 2.7: Tactile Switches in Housing

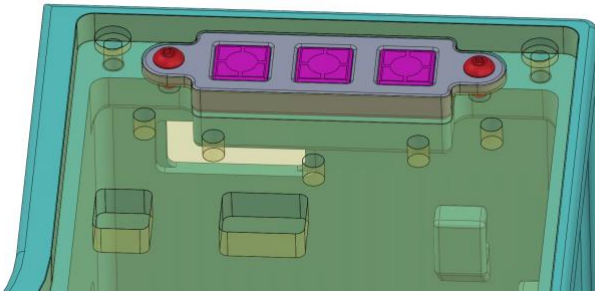


Figure 2.8: Flush Mount Switches

2.4 LCD Friction Hinge and Frame

The viewing angle of the LCD can be adjusted due to a friction hinge mounted to the top plate of the enclosure by (4) M3 machine screws. The friction hinge will provide 1.2 in.-lbs. of holding resistance, allowing the LCD to remain at the user placed angle. The friction hinge will be mounted to the LCD mounting frame, in which the LCD will be located. This design allows the LCD to be placed in a stowed configuration, decreasing its overall height. The piano hinge type mounting bracket increases its ability to withstand off axis loads it may experience if it gets bumped into.

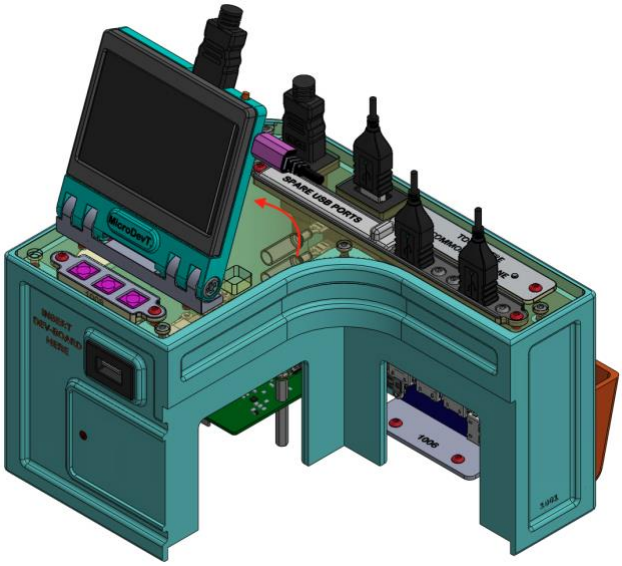


Figure 2.9: Friction hinge Location

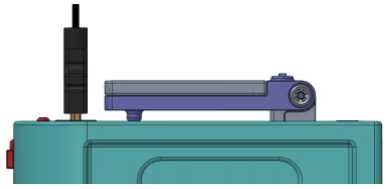


Figure 2.10: Screen in stowed position

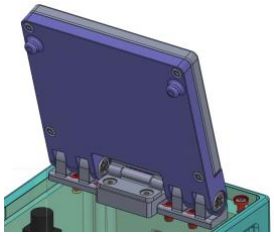


Figure 2.11: Screen in Use

2.5 LCD USB-A and HDMI Ports

The LCD is powered through a USB-A to USB Micro connection and the display is driven using an HDMI connection. Both of these connections are made via the top cover of the large enclosure which contains two cutout features where the snap-in USB-A and HDMI ports are located.

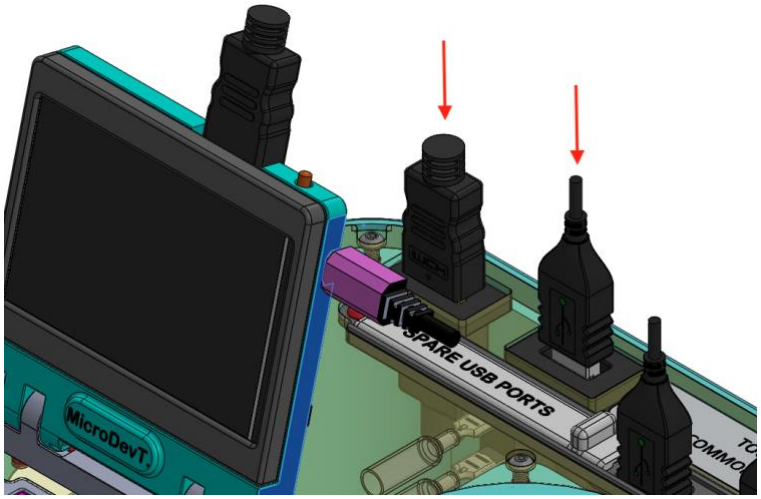


Figure 2.12: HDMI / USB-A Ports

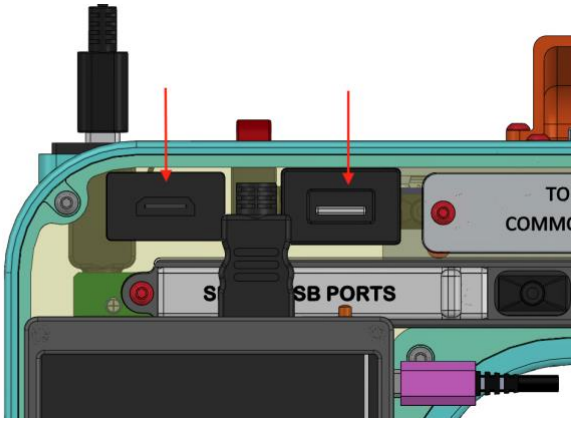


Figure 2.13: Snap-in Ports Inserted

2.6 Access Panel and ESD Grounding Plate

There may arise a need for the administrator to gain access to the functions of the Raspberry Pi computer. For such an event, a keyboard and mouse may be necessary. To account for this possibility, the large enclosure contains a four-port USB-A hub with two spare USB-A ports. These ports will be accessible by removing the small access panel mounted to the top cover by a single screw. There is a slight taper to the access panel to aid in removal.

There is also an ESD grounding plate located above the access panel. This plate is 14 gauge aluminum which is tied to the ground plane of the "Little Foot" PCB. This plate provides a location for the user to ground themselves before interfacing with the device.

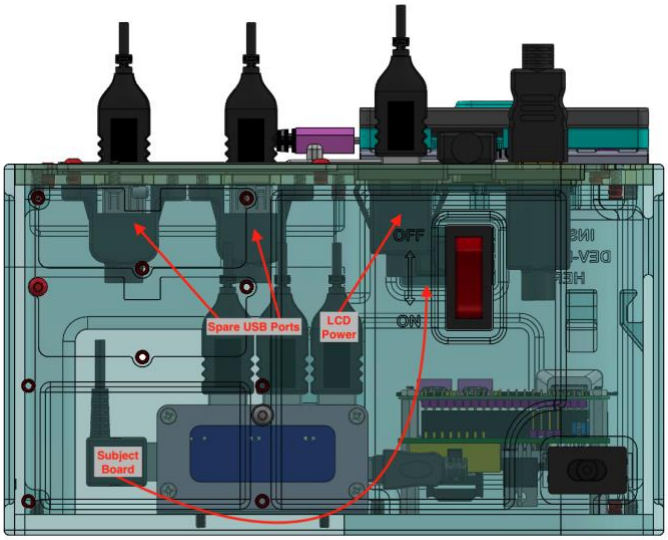


Figure 2.14: USB-A Hub port allocation

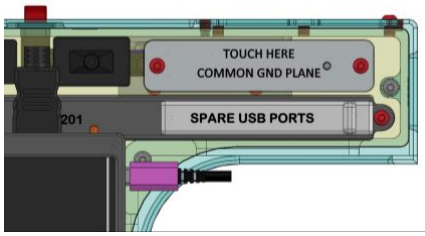


Figure 2.15: Access panel and ESD grounding plate

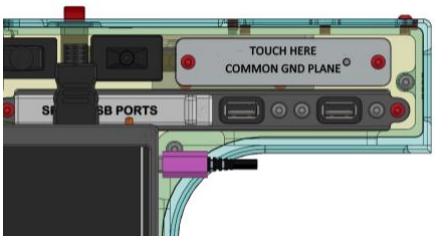


Figure 2.16: USB cover pushed to the left exposing spare USB ports

2.7 Four-Port USB Hub

The four-port USB hub is mounted to the interior side of the back wall, of the large enclosure. This USB-A hub is powered from the Raspberry Pi Zero 2W's Micro USB port.

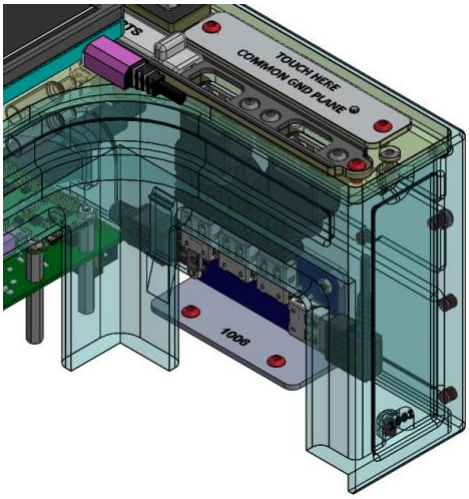


Figure 2.17: USB Hub Location

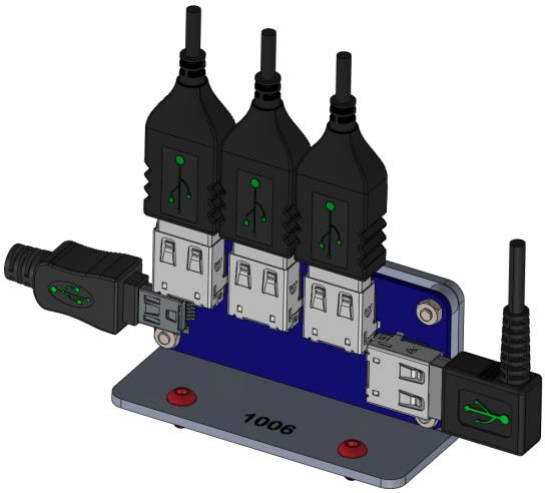


Figure 2.18: USB hub mounting bracket

2.8 Carry Cutout

A small finger cutout is located on the left side of the enclosure to increase portability. This feature also serves as a vent for heat dissipation while the Raspberry Pi is powered on.

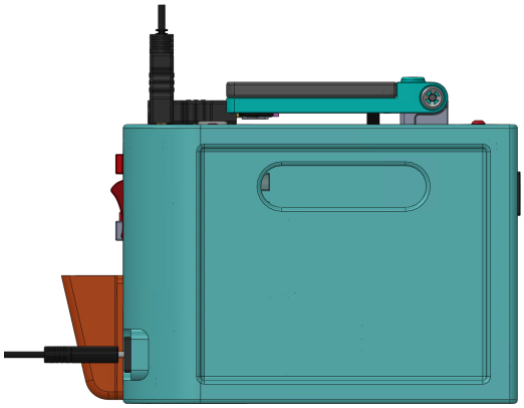


Figure 2.19: Finger Cutout

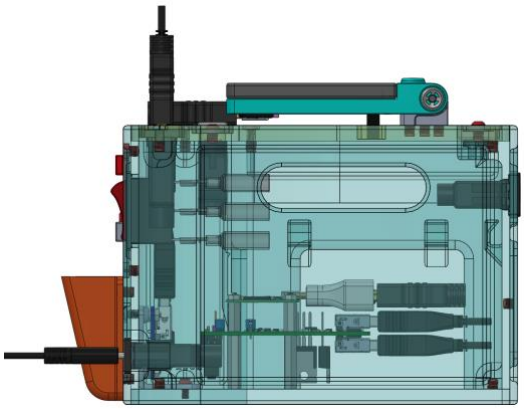


Figure 2.20: Interior component locations in relation to finger cutout

2.9 Main Power Switch and USB C Power Port

The main power toggle switch is mounted via a cutout feature located on the back of the enclosure. This feature satisfies specification 2.00.1 which states: "The device must use a toggle switch to control the main power supply".

Another small cutout feature is located down and to the right of the main power switch. This feature is used to mount the snap-in USB-C power adapter port. This feature satisfies specification 2.01.1 which states: “The device must use a 120V AC”.

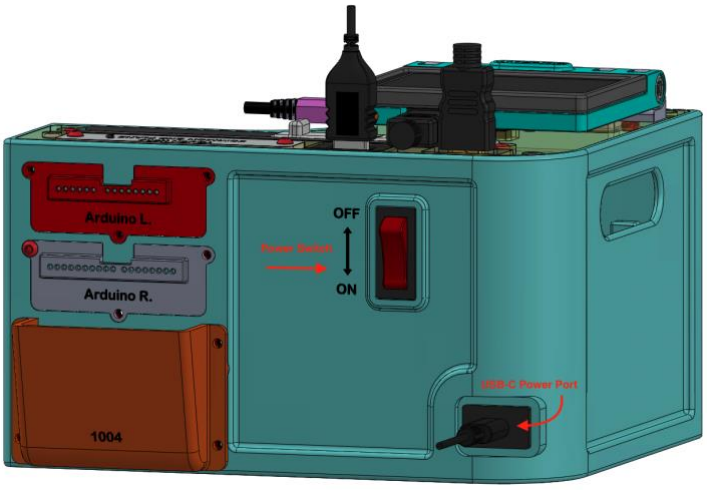


Figure 2.21: USB C power port and switch

2.10 Device Storage

There are locations on the back of the device for storing the header covers not currently in use. Additionally, there is a small pocket to store the header extensions or spare parts.

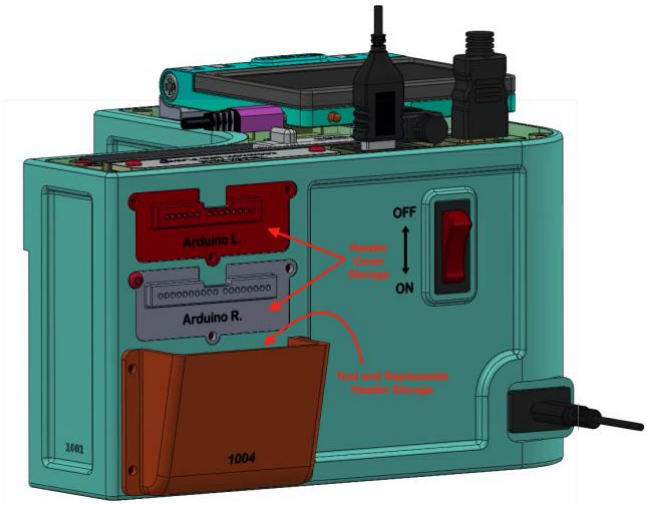


Figure 2.22: Storage features

3.0 Small Enclosure Assembly

The smaller enclosure houses two sub assemblies which include: the “Tool-Pack” assembly and “Ejection Crank” assembly. The housing of this enclosure has three notable features: dovetail joints, interior cutouts, and ejection

crank handle cutouts.

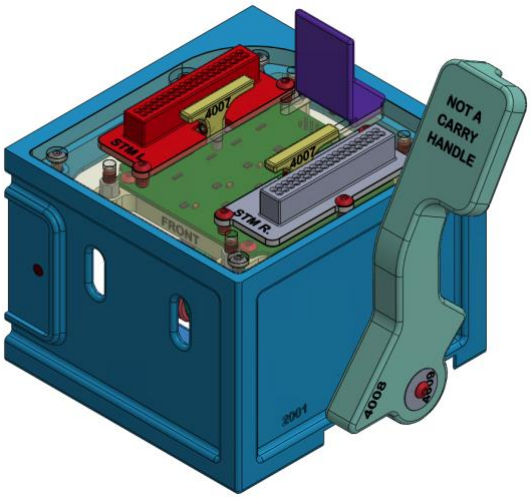


Figure 3.1: Small Enclosure Assembly

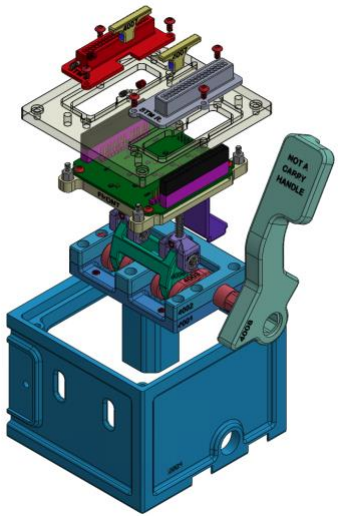


Figure 3.2: Exploded View

3.3 Ejection Crank and Hard-Stop Features

The ejection crank extends through a cutout feature on the right side of the enclosure which allows the handle to be mounted on the exterior of the enclosure. There are two other cutout features on this side of the small enclosure which allows the large ABS bottom plate to extend out in the form of two tabs. These two tabs have small, replaceable hardstop blocks mounted on top.

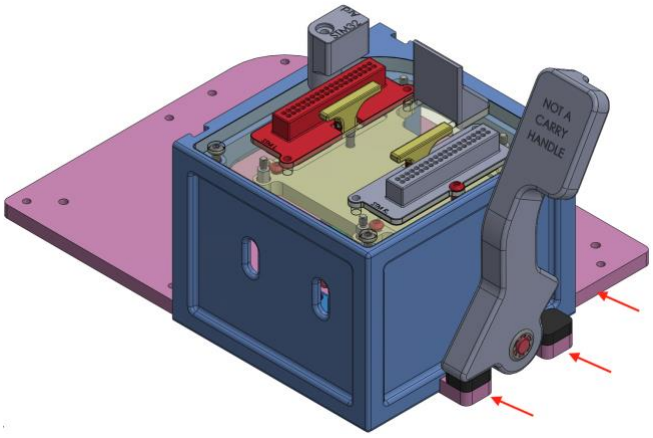


Figure 3.5: Hardstops

4.0 Tool Pack Sub-Assembly

Although the Arduino and STM subject boards are different in size and header configuration, they will share the same tool pack with the exception of their header covers. This means the Arduino and STM boards can be mounted to the fixture by changing out a few removable components. The tool pack highlighted in **Section 4** includes the PCB mounting plate, PCB header interfaces, detachable headers, header covers, and alignment posts.

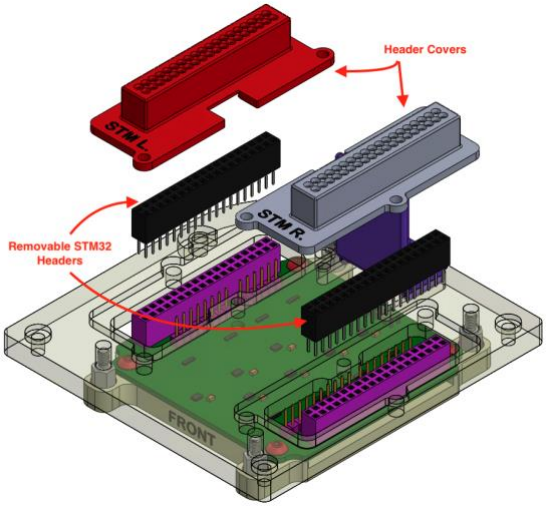


Figure 4.1: Tool Pack

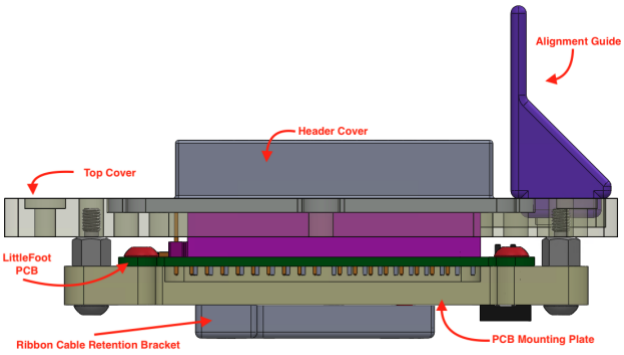


Figure 4.2: Tool Pack from the Side

4.1 PCB Mounting Plate

The bottom layer of the tool pack is the PCB mounting plate. This plate allows the PCB to be mounted to a non conductive CNC machined surface. The side cutout features on this plate give room for the soldered joints below the PCB for the through hole headers. This plate will be mounted to the "Top Plate" of the tool pack via the standoffs seen at the corners of the mounting plate, which use the holes highlighted in pink.

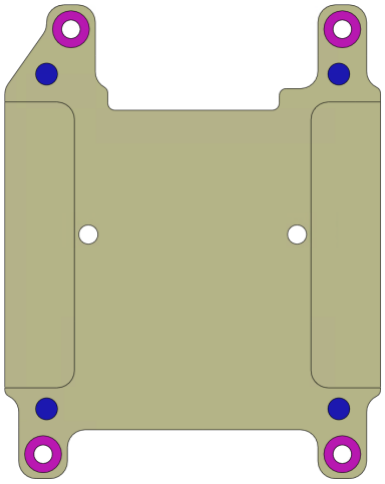


Figure 4.3: PCB Mounting Plate

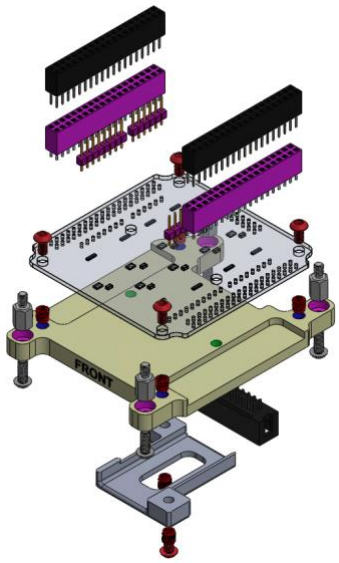


Figure 4.4: PCB mounting plate with PCB and headers

4.2 Little PCB with Removable Headers

The little PCB contains the header pin interface to the subject board and the ribbon cable harness header pin interface which routes signals to and from the big PCB containing the ICs for measurements. The little PCB and its mounting plate are common between the subject boards. The only difference in these configurations is which

header extensions have been mounted. Below shows the two configurations.

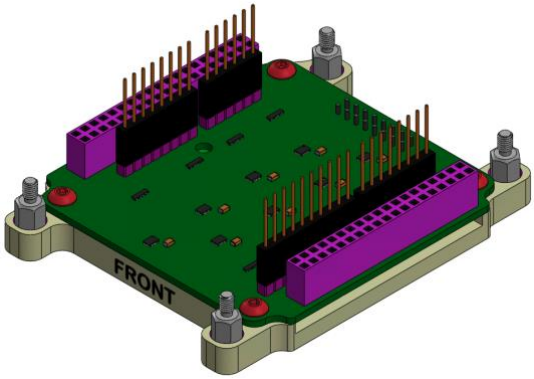


Figure 4.5: Little PCB in **Arduino Configuration**

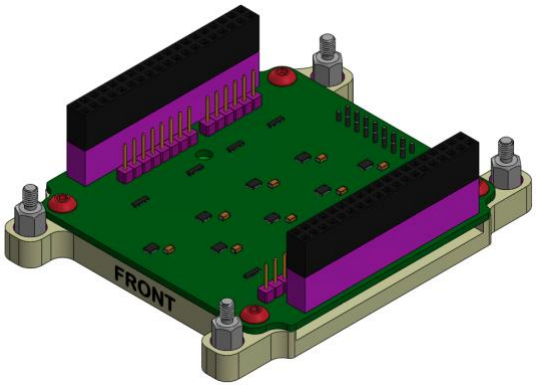


Figure 4.6: Little PCB in **STM Configuration**

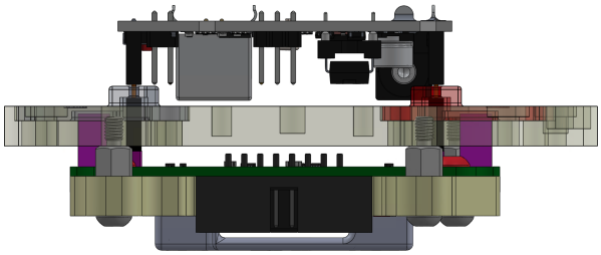


Figure 4.7: Rear view of assembly when **mounting an Arduino board**

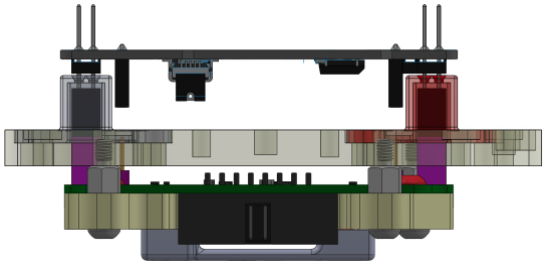


Figure 4.8: Rear view of assembly **when mounting a STM board**

4.3 Small Enclosure Top Plate

The component above the PCB is the small enclosure top plate. This component has two relatively large cutout features in the center where the header covers are located. These cutouts allow the PCB headers to pass up through the top plate and also give a surface for the header covers to seat.

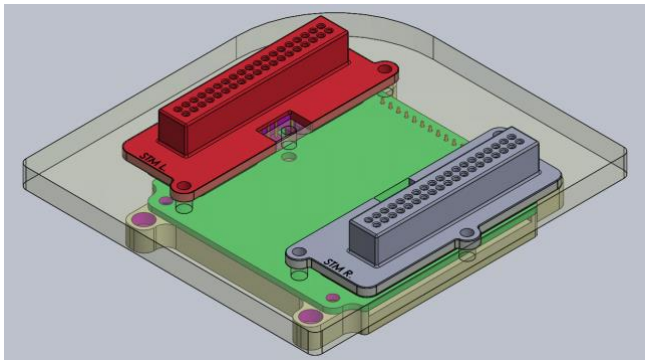


Figure 4.9: Top plate in the STM configuration

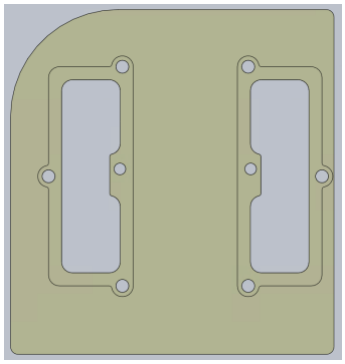


Figure 4.10: Small enclosure top plate cut out features

4.4 Removable Header Covers

The header covers are used to secure the replaceable headers in their positions. These are removable via three screws located around its perimeter. The header covers contain a cutout feature to allow the push rod hats to move up and down with the ejection crank assembly push rods.

There are four header cover configurations: Arduino left and right covers and STM left and right covers. When switching the device over from testing an Arduino to STM (or vise-versa), the user must first remove the header covers, swap out the header extensions, and replace the header covers for the correct board.

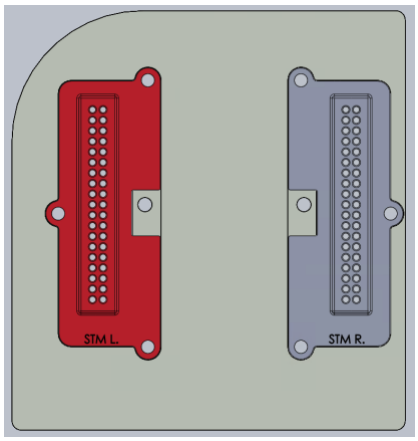


Figure 4.11: STM header covers

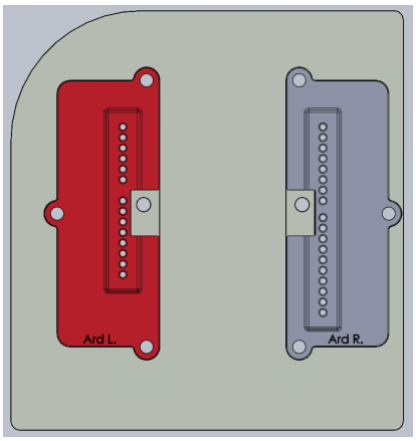


Figure 4.12: Arduino header covers

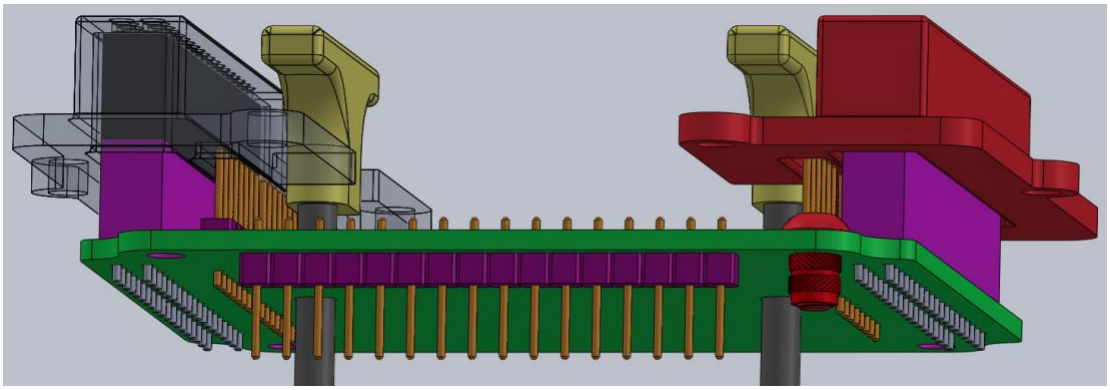


Figure 4.13: Header covers with little PCB and push rods/hats, in the STM configuration

4.5 Alignment Posts

An alignment guide is mounted to the top plate which touch off on the far side of the subject boards. This alignment component has two adjustments, one for the arduino, and another for the STM32. The figure below shows the STM32 overlaid with the Arduino to demonstrate the adjustments of these components.

To adjust, loosen the top screw and slide the guide all the way to the back. Mount the subject board on the header interface and slide the guide forward until its flush with the edge of the subject board. Re-tighten the screw when in place.

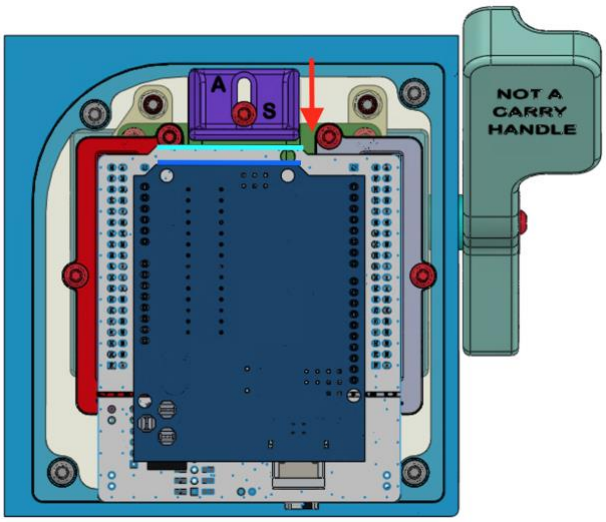


Figure 4.14: Alignment guide with STM and Arduino overlay

4.6 Pressing Locations on the Subject Boards

The push rod hats contact both subject boards just inside their Arduino female header pins. This location works well since the STM also has arduino headers mounted to the board. The location of the hats are centered on the STM headers. This is done to reduce unnecessary torque on the push rods, which may result in binding during operation. The exact location of these contact points can be seen below on both the STM and Arduino.

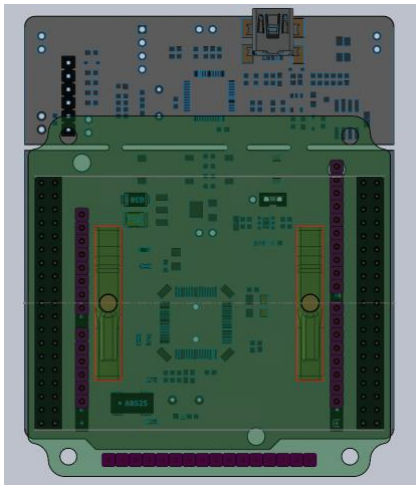


Figure 4.16: Push rod hat contact locations on the STM board

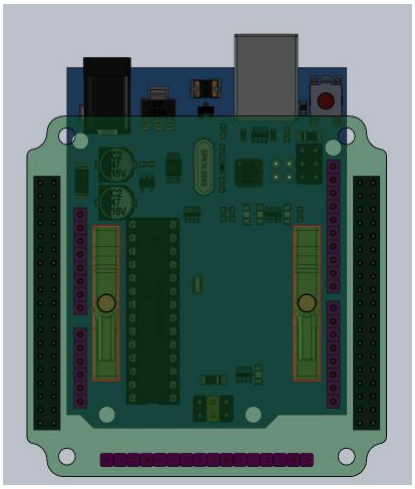


Figure 4.17: Push rod hat contact locations on the Arduino board

5.0 Ejection Crank Sub-Assembly

The ejection crank assembly provides mechanical leverage to aid the operator in the removal of the subject board from the header pin interface. The handle is directly mated with the crankshaft so when the operator rotates the handle in the clockwise direction, the crankshaft assembly drives push rods upward. The push rods are guided linearly upwards pushing the subject board off the header interface. This assembly satisfies specification 1.03.2

which states: “The device must use a mechanism to easily and safely remove the subject board without causing damage”.

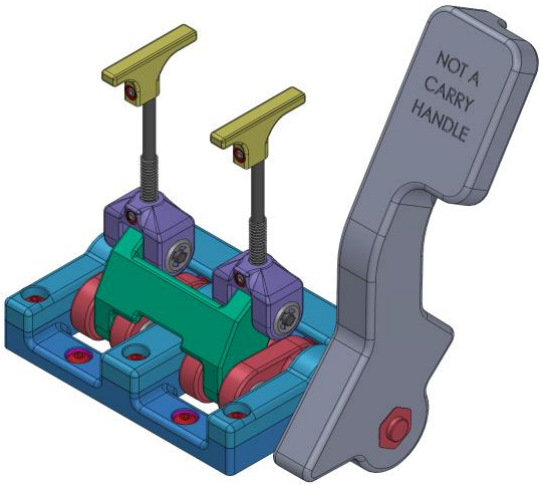


Figure 5.1: Ejection crank assembly

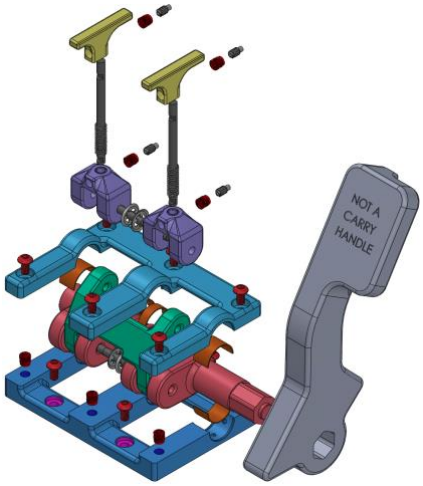


Figure 5.2: Exploded view

5.1 Crankshaft Base Plate

The ejection crank assembly is mounted to the large base plate of the fixture (not seen here). This component secures the ejection crank assembly in place and contains the lower races which the crankshaft rotates on.

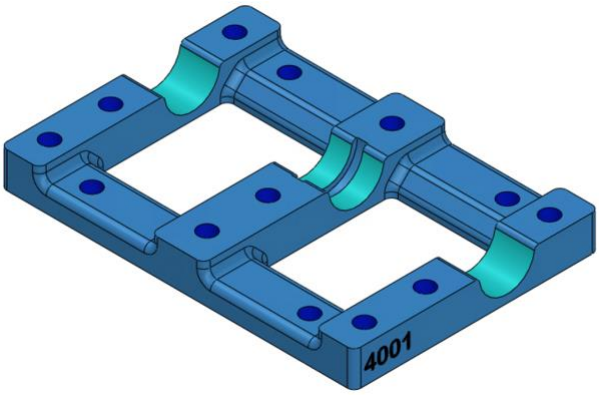


Figure 5.3: Features of the cranks base plate

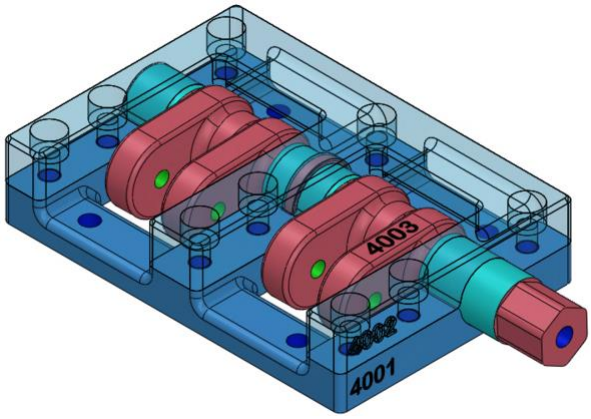


Figure 5.4: Crank base with crankshaft and cover

5.2 Crankshaft Cover

The crankshaft cover contains the upper crankshaft races and is fastened to the crank base. This component works with the crankshaft base plate to locate and secure the crankshaft in place.

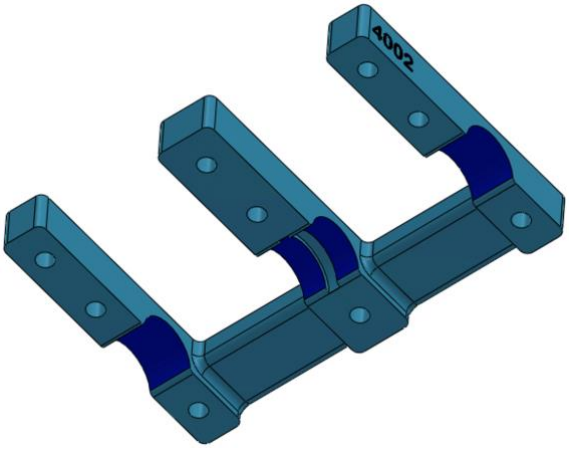


Figure 5.5: Crankshaft cover features

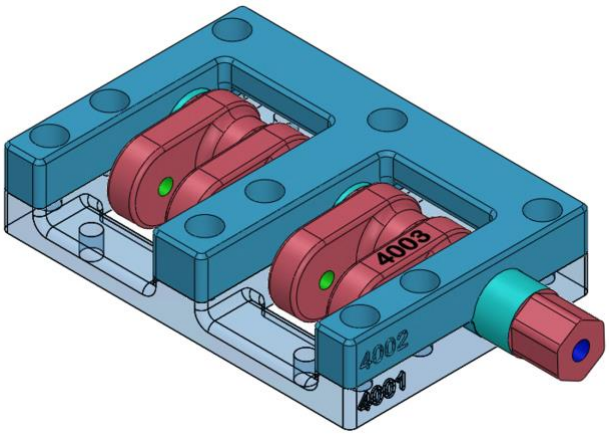


Figure 5.6: Crank cover with base plate

5.3 Crankshaft

The crankshaft transmits the operator's rotational motion into vertical linear motion. It's a single component with chamfers around high stress areas to increase strength. There's two round extruded guides on the crankshaft to position it in the crankshaft base plate and cover housing. The end of the crank has an extended segment to allow the handle to be pressed into place. This section of the crank is seen as having a hex key feature to prevent the handle from rotating on the shaft.

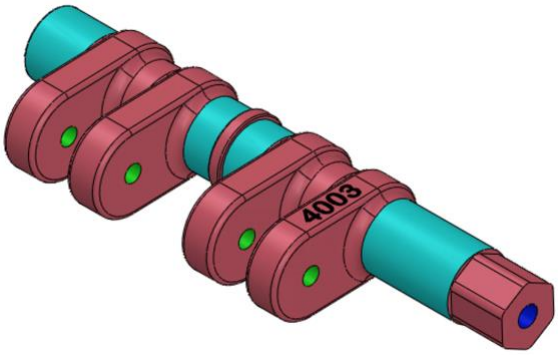


Figure 5.7: Crankshaft Features

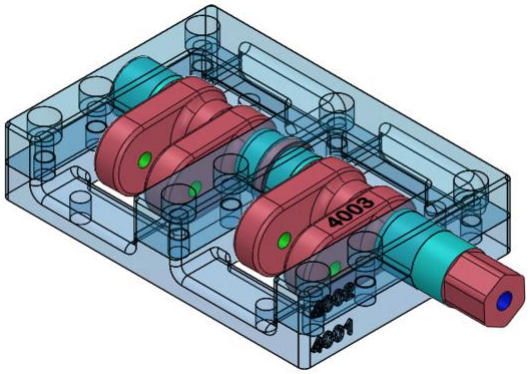


Figure 5.8: Lower ejection crank assembly

5.4 Crank Linkage

In this design, two chain link type components have been tied together to add additional rigidity to their small stature. This component seen in **Figure 5.9** provides the second and third pivot points in this assembly. The link is driven by the crankshaft, pivots around ⅛" stainless steel rod stock, and drives the push rods upward. The pin components are held in place by pressing on metal retaining clips.

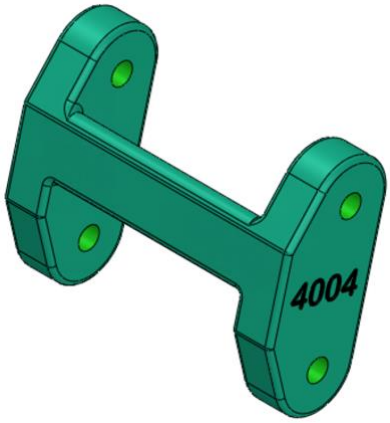


Figure 5.9: Connected Link Component

5.5 Push Rod Foot / Push Rod / Push Rod Hat

The push rod is indexed on both ends and is pressed into the push rod foot and hat. The index features prevent the pushrod hat from rotating to an undesirable direction, which could damage the subject board. The push rods travel vertically, guided by holes through the PCB mounting plate and the extruded guides in the header pin covers. An adhesive strip is inserted into the cutout of the push rod hat to prevent damage to the subject board.

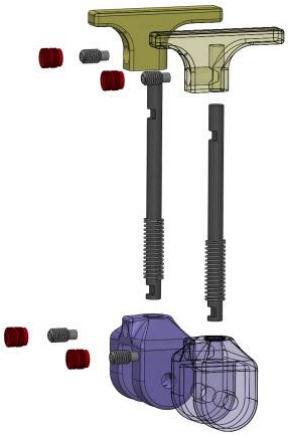


Figure 5.10: Exploded view

Figure 5.11: Set screw retention system

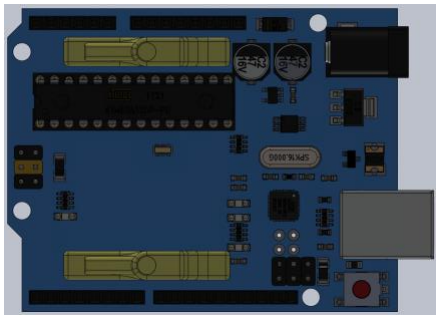


Figure 5.12: Push rod hat contact locations on Arduino

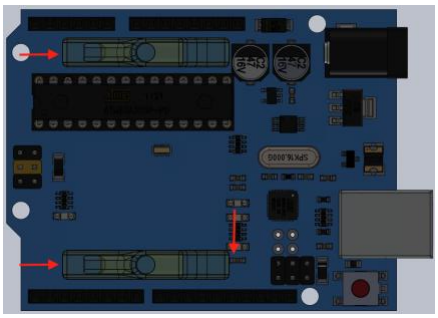


Figure 5.13: Push rod hat contact location clearences

5.6 Return Springs

Return springs are placed over the push rods and press down against the PCB mounting plate. These springs return the crank assembly and crank handle to the down position. This prevents the ejection crank assembly from undesirable movement during operation and transportation.

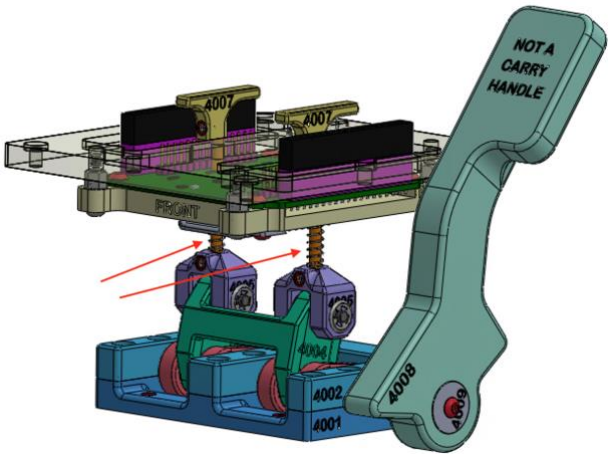


Figure 5.14: Return springs in relation to assemblies

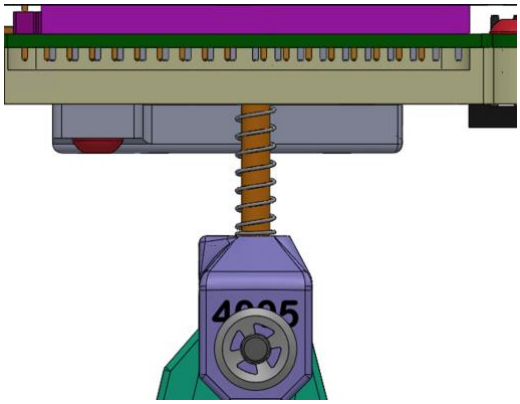


Figure 5.15: Push rod foot with spring perches, compression spring, and PCB mounting plate

5.6 Crank Handle

The handle has a hex cutout feature which is used to mate with the crankshaft. Again, this is designed to prevent the handle from rotating on the crankshaft. The crank handle is designed with enough length to provide a significant mechanical advantage for ejecting the subject board from the header interface.

Figure 5.16: Crankshaft Handle

Figure 5.17: Crankshaft Handle

5.5 Crank Assembly in Return and Advanced Positions

The crank assembly is offset from the axis of the push rods, as seen below. This is to reduce sideloading the push rods, reducing rod binding that may occur when the assembly is advanced. The angle from the top pin to the bottom pin was kept to 20 degrees to ensure the assembly motion will be smooth and energy transmition will be directed as upward as possible.

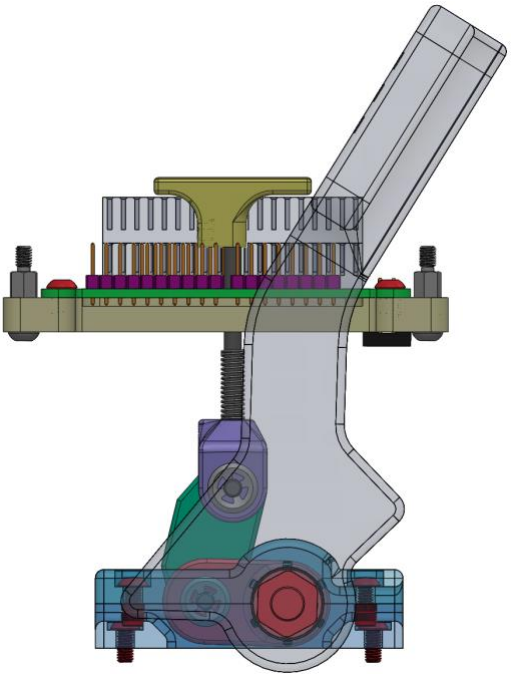


Figure 5.18: Linkage in return position

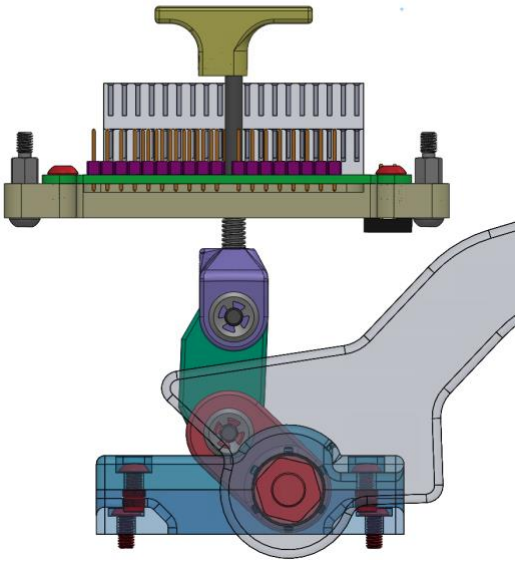


Figure 5.19: Linkage in advanced position

6.0 Enclosure Validation Methods

The table below outlines the methods of validation for each specification of the enclosure.

Table 5. Enclosure validation methods

Specification	Method of Validation
1.00.1	The size of the fully assembled enclosure will be confirmed using a tape measure.
1.01.1	The weight of the fully assembled enclosure will be confirmed using a scale.
1.01.2	The alignment posts of the enclosure will be calibrated to align a subject board to the header pins. The subject board will then be removed and the enclosure will be transported by hand approximately 40ft. Once placed back on a flat surface, the calibration of the alignment posts will be confirmed by attempting to interface the subject board back on the enclosure without re-configuration.
1.02.1	Each component and cable will be visually inspected to confirm that it is securely fastened to the enclosure.
1.03.1	The enclosure will be visually inspected to confirm the alignment post is present.
1.03.2	The enclosure will be visually inspected to confirm the ejection crank assembly has been installed. A subject board will be placed on the header interface and removed using the ejection crank assembly 10 times. After this cycle, the subject board will be visually inspected to confirm that no damage has been inflicted to the board. Three different operators will conduct this test to ensure no hand blisters or cramps result from operation.
1.04.1	The print settings will be visually confirmed by the 3D print operators before the start of each print.

2.00.1	The enclosure will be visually inspected to confirm the presence of the main power toggle switch. The switch functionality will be confirmed by applying 5V and measuring the pin to pin voltage for each position of the switch.
2.04.1	The enclosure will be visually inspected to confirm the presence of the removable header pins to which the subject board interfaces.

Hardware Design

The device uses two custom built PCBs in order to interface with the Raspberry Pi and the subject board. The first custom PCB, named BigFoot, contains all of the ICs used for testing and interfacing with the Raspberry Pi. The second PCB, named LittleFoot, contains the multiplexers used to isolate each pin on the subject board as well as replaceable header pins for interfacing with the subject board. Having two PCBs allows for a more modular system and simpler interface with the mechanical fixture. **Figure 7.0** shows how the PCBs are interfaced with the rest of the device.

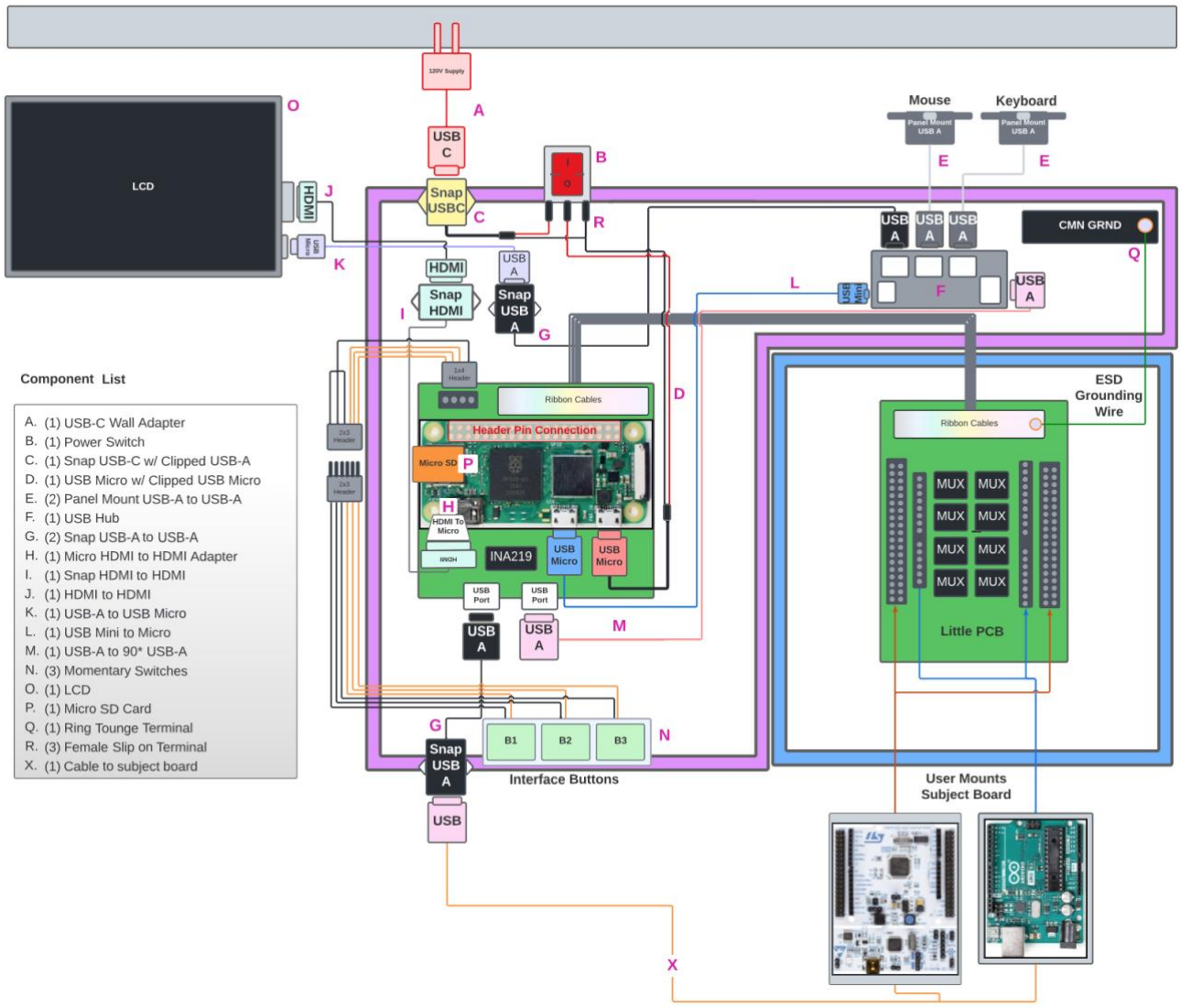


Figure 7.0: Device Connection Interface

An external switch connected to a wall outlet is used to supply power to the Raspberry Pi, which is mounted directly onto the BigFoot PCB. The Raspberry Pi's USB power is then routed into a USB Hub which is used to supply the subject board and any extra peripherals the device may need, such as a keyboard and mouse. A 2x8 ribbon cable is used to transfer power and data from BigFoot to LittleFoot. Three GPIO signals are also routed from BigFoot to three external buttons that are used to control testing and view test results.

7.0 BigFoot PCB

Schematics and PCB layouts for both boards were done in Altium. The BigFoot PCB contains several different Integrated Circuits (ICs) that are used for testing the subject board. These ICs include an Analog-to-Digital Converter (ADC), a Digital-to-Analog Converter (DAC), and a current measurement IC. These ICs can be seen in **Figure 7.1**, which shows the overall schematic of the BigFoot PCB.

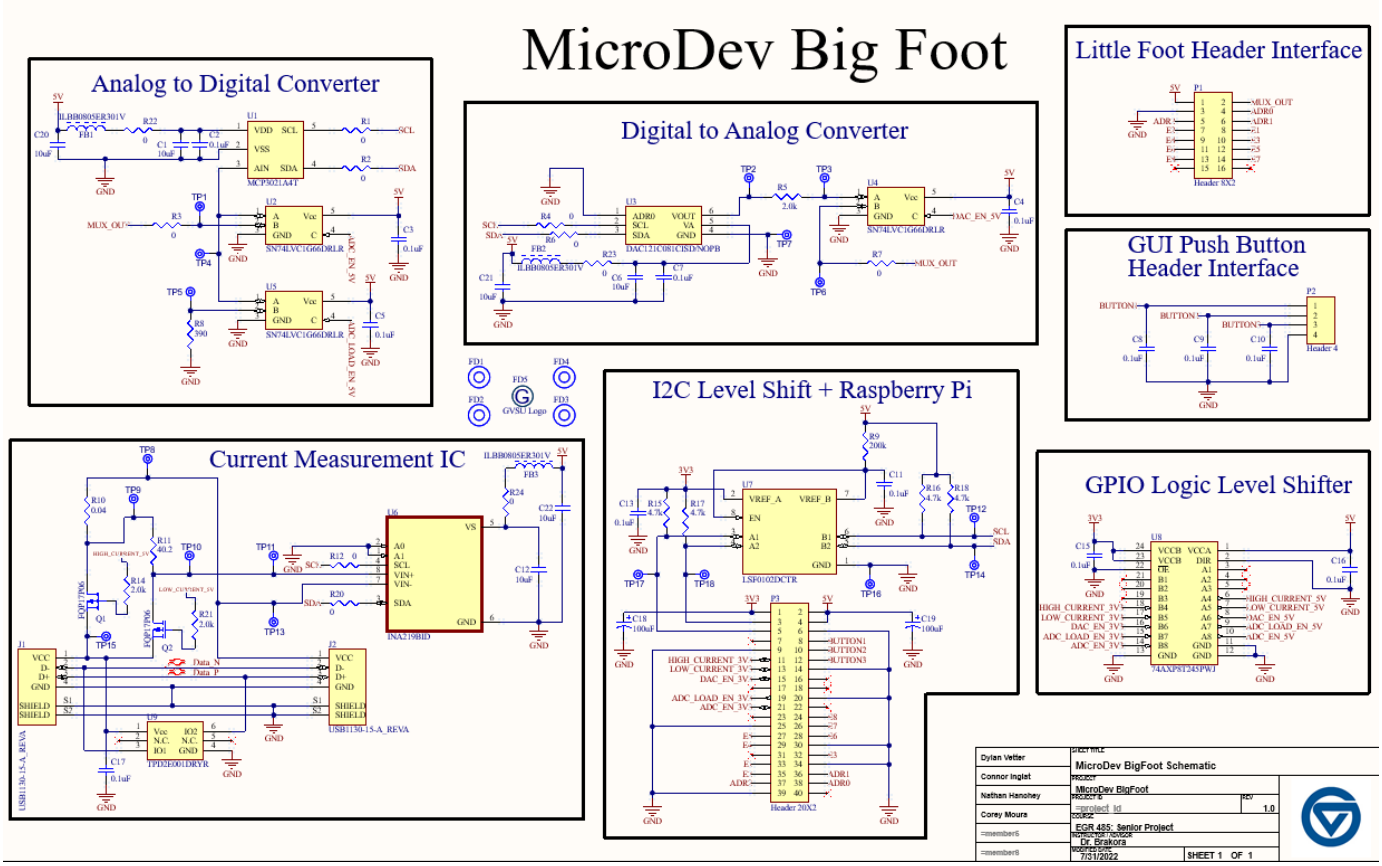


Figure 7.1: BigFoot Schematic Overview

The ADC, DAC, and current measurement IC are controlled by the Pi using I²C. The schematic also contains header pins for interfacing with the Raspberry Pi as well as a logic level shifting IC for I²C communication between the Pi and the ICs. The ADC circuit tests the subject board's voltage supply pins, GPIO output voltage under load, internal resistance, and GPIO pull-up voltages. The DAC circuit tests the subject board's logic levels, ADC, and GPIO wakeup from sleep modes. The current measurement IC tests the subject board's current draw (and therefore power consumption) during normal operation and in sleep modes. BigFoot also contains header pins for routing GPIO signals to the external push buttons on the device.

7.1 LittleFoot PCB

The LittleFoot PCB contains all of the header pins and multiplexers(MUXs) needed to interface with the subject board. The Multiplexer shown in **Figure 7.2** is the TMUX1208QRSVRQ1, an 8-channel bidirectional analog switch.

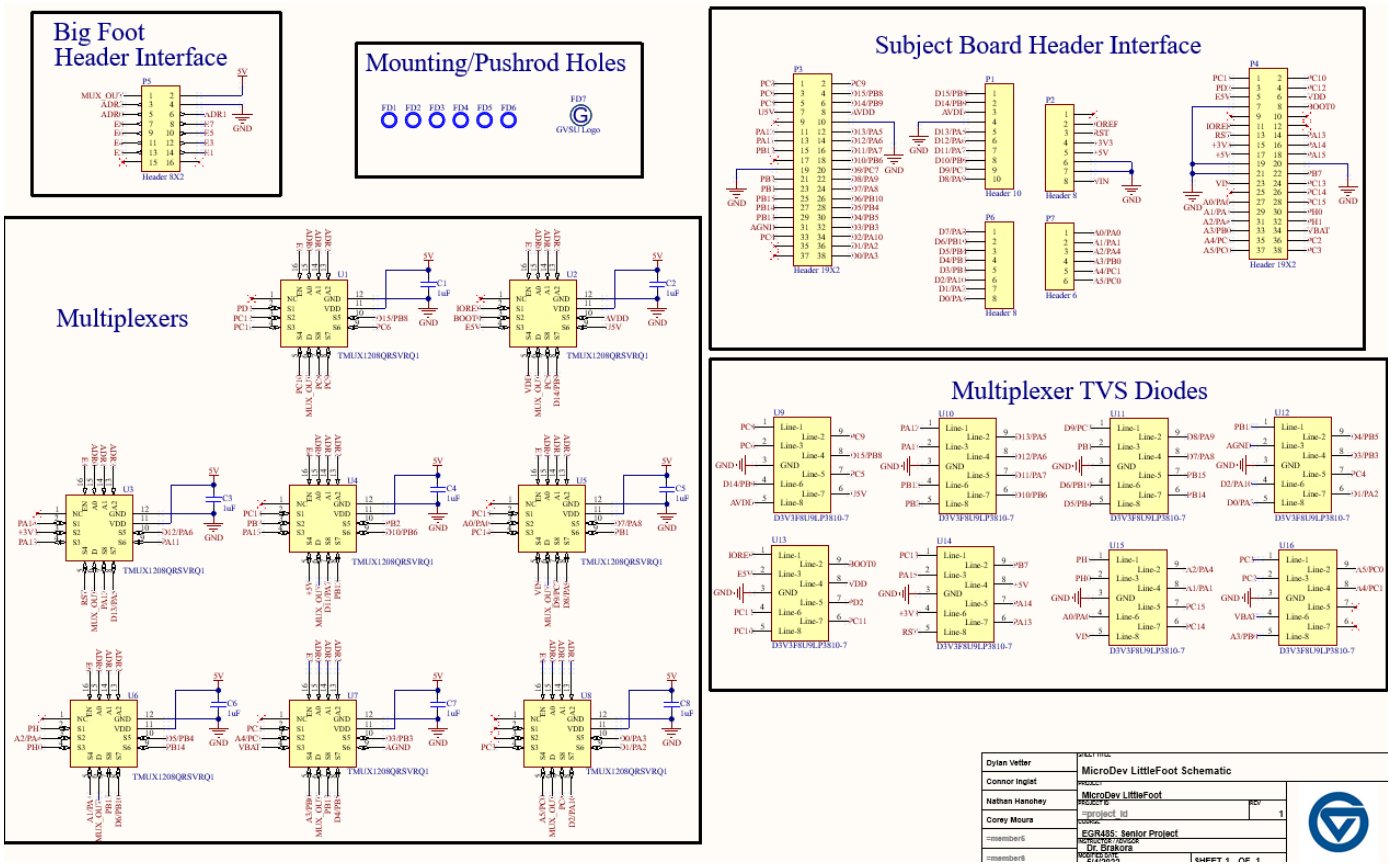


Figure 7.2: LittleFoot Schematic Overview

The MUX has a low 5-9Ω on-resistance and supports bidirectional input/output. Currents through the MUXs will be limited to below 30mA, as that is the max rating of the IC. The A0, A1, and A2 pins are the input select pins that control which of the eight I/O pins are connected. Each of the MUX's eight I/O pins are connected to a pin on the subject board via the header pin interface that the user connects their subject board to. The enable pin (E) controls whether the I/O is open or shorted. The common input/output (D) is labeled MUX_OUT.

The MUXs are used to connect every pin on the subject board to the BigFoot PCB in order to be used for testing. Raspberry Pi GPIO pins are used to control the state of each MUX. Multiple MUXs are placed in a cascaded configuration in order to increase the number of channels available while reducing the number of GPIO control pins used. The A1, A2, and A3 input select pins are shared between each of the MUXs. The enable (E) pins of each MUX however are controlled by individual GPIO pins on the Raspberry Pi. Control of the E pins and address pins allow for each of the MUXs' common output D to be tied together. Through proper control of the MUXs, it can be ensured that only one input and output is connected within the cascaded network. Tying the outputs D together gives the advantage of only having to route one trace into the ICs used for testing.

7.2 ADC Circuit

Table 3. shows the requirements and specifications that are met by the ADC.

Table 6: Voltage Measurement Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
2.05	The device must **accurately measure** the subject board **voltages** under test	2.05.1	The device must implement hardware that measures voltage with an accuracy of 0.1V within the range 0 to 5.5 V unless otherwise specified
3.02	The device must test and record each GPIO pin's output voltage under **load sourcing**	3.02.1	The device must test that the pin's output voltage remains above a configurable threshold
3.03	The device must test and record each GPIO pin's internal resistor value	3.03.1	The device must test that the calculated internal resistance of each GPIO pin is within the manufacturers range
3.04	The device must test and record each GPIO pin's internally resistive **pull-up voltage while unloaded**	3.04.1	The device must measure voltage from each GPIO to GND while pin is configured as pull-up
		3.04.2	The device must measure and test each GPIO pin's voltage while configured as pull-up to be within a configurable range
1.04	The device **must have an enclosure that can withstand the rigors of daily use**	1.04.1	The device must have 3D printed parts with an infill of at least 30%
2.00	The device must utilize a physical **switch to control main power**	2.00.1	The device must use a toggle switch to control the main power supply
2.04	The device must have a **servicable hardware interface** to the subject board	2.04.1	Must have a header that connects to the I/O pins of the subject board

The ADC seen in **Figure 7.3** is the MCP3021A4T, a 10 bit precision analog to digital converter with an I^2C interface.

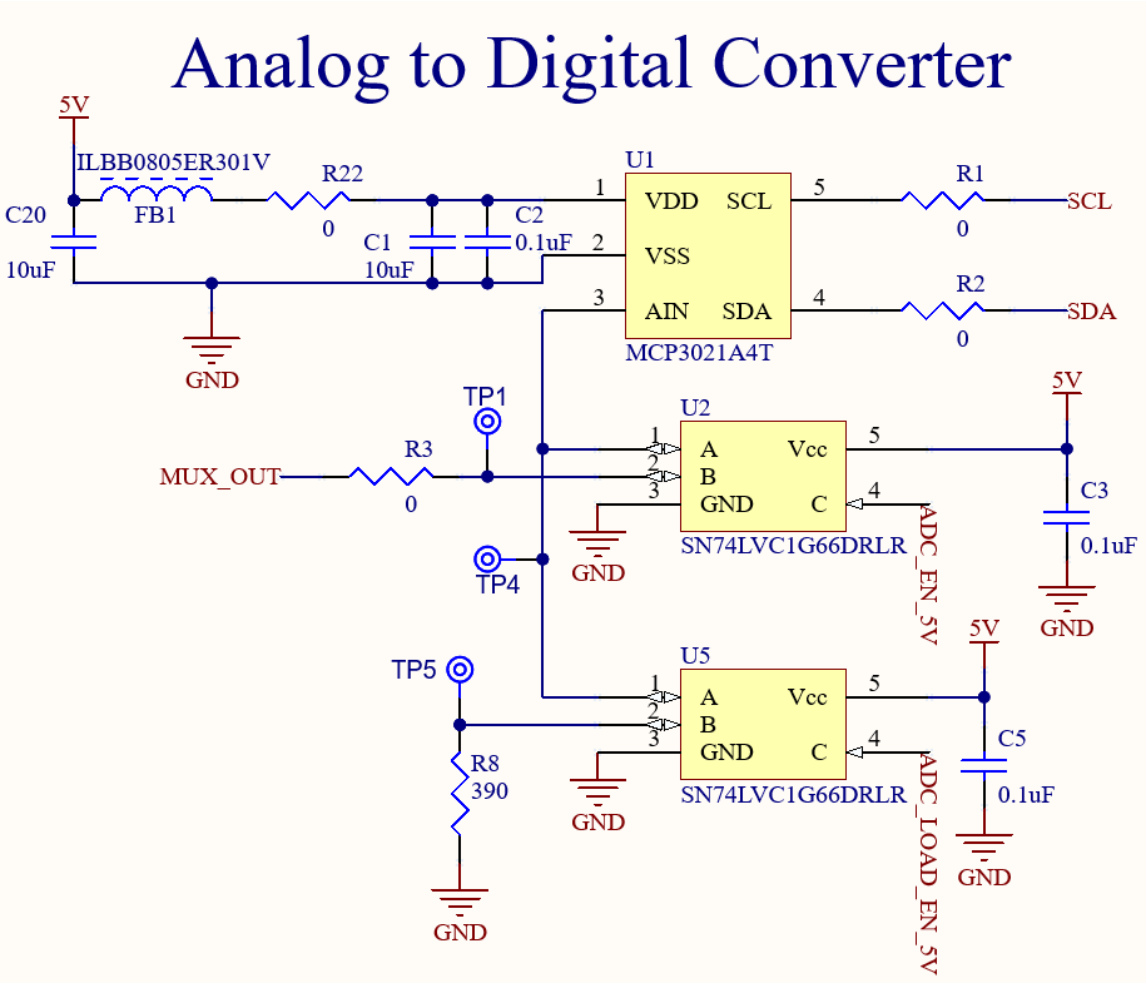


Figure 7.3: Analog to Digital Converter Integrated Circuit

The ADC's resolution can be calculated as:

$$Resolution = \frac{Full\ Scale\ Range}{2^N} = \frac{5.3 - {}^{-}0.3}{2^{10}} = 5.46\ mV$$

The ADC has an offset error of ±0.75 LSB that can be accounted for in software. By applying 0V to the input, the output will then be the offset error of the ADC. This value is subtracted from all future ADC measurements. Typical gain error of the IC is -1 LSB, which is also accounted for in software. The maximum voltage of the full scale range, or the maximum voltage before the output saturates, is applied to the input. The deviation from the actual output to ideal output is the gain error, which is used to calibrate the ADC in software by scaling the ADC output values up or down depending on the gain error.

Bypass capacitors are placed in parallel with the supplied voltage to the ADC, as recommended by the datasheet in order to eliminate high frequency noise and to supply momentary bursts of extra current required from the supply when the device is converting. Additionally, a ferrite bead is placed in series with the supplied voltage to further eliminate unwanted high frequency noise.

The ADC communicates its voltage measurements to the Pi via the SDA line. The Pi also uses the SDA line to trigger the ADC to take a new measurement or to power down. The switches U2 and U6 are the SN74LVC1G66DRLR, a 1:1 bidirectional analog switch with 5.5-10Ω resistance. Current through the switch is limited to below 50mA since that is the maximum rating of the switch. The switches are controlled by Pi GPIO pins. When the switch U2 is closed, the ADC is connected to the circuit so that voltages may be measured. The configuration to measure the subject board's supply pin voltages as well as each GPIO pin's unloaded voltage when configured as pull-up or pull-down is shown in **Figure 7.4**.

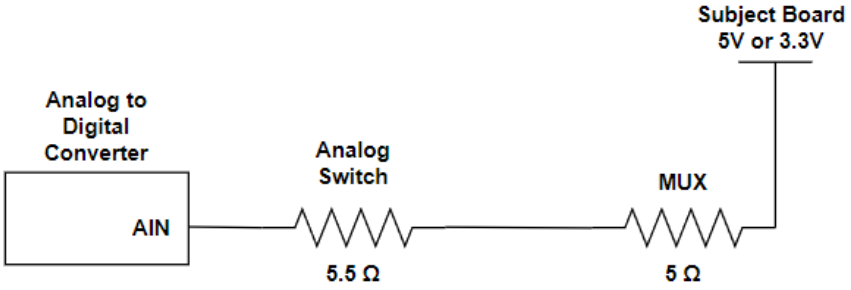


Figure 7.4: Supply Pin Voltage Measurement Configuration

The ADC's AIN input channel is assumed to have an infinite impedance and is treated as an open circuit. Therefore, the switch and MUX will have no effect on the voltage measurement as no current is flowing through either component.

When the switch U6 is also closed, the 390 Ohm resistor connected to ground is connected with the ADC input line. In this configuration, the ADC is capable of measuring the subject board's GPIO output voltage under load, as seen in **Figure 7.5**.

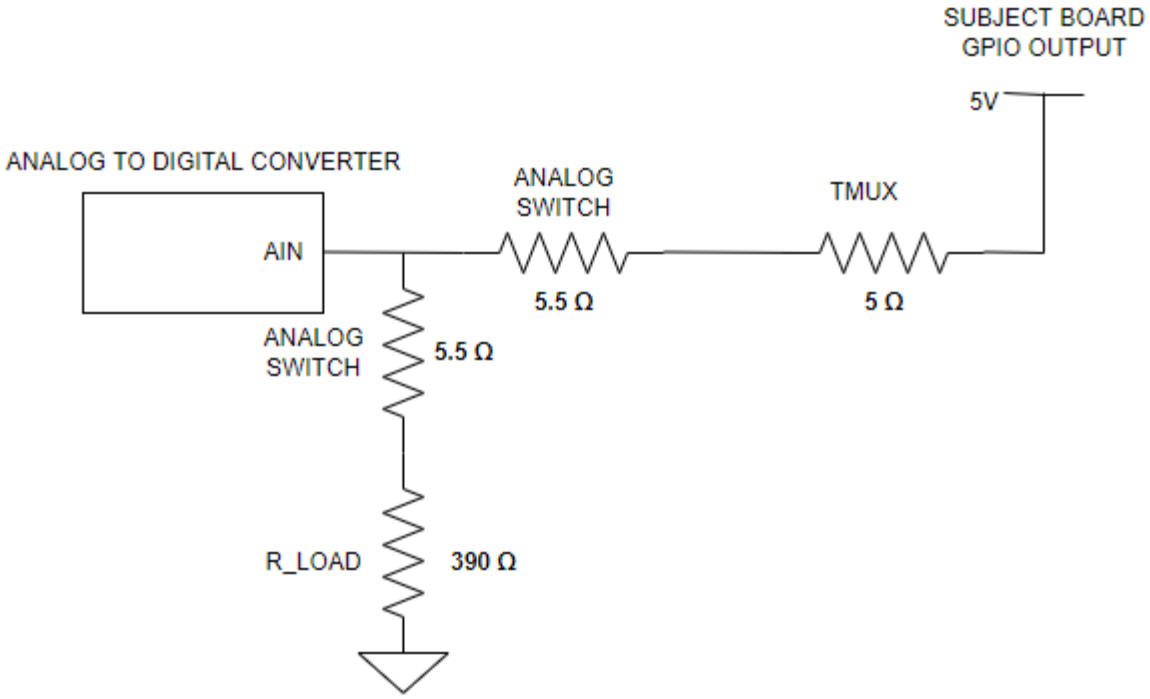


Figure 7.5: GPIO Output Voltage Under Load Measurement Configuration

GPIO pins on the arduino are rated for 40 mA of output current and 25mA on the STM Nucleo. It is recommended to limit the output current to around half of the rated value in order to avoid damage to the pin. For this reason, a 390Ω load resistor was chosen.

In this configuration, the switches and MUXs will have an impact on the voltage measurement due to the on resistance of each component. The minimum on resistance of the MUX is 5Ω while the maximum on resistance of the analog switch is 5.5Ω. Assuming these minimum values and a 10% tolerance on the load resistor, the voltage at the ADC input channel can be calculated as:

$$R_{series} = 5 + 5.5 + 5.5 + 351 = 367\ \Omega$$

$$i_{out} = \frac{5V}{367\ \Omega} = 13.62\ mA \approx \frac{1}{2} i_{rated}$$

The expected voltage reading of the ADC is then calculated as:

$$V_{AIN} = V_{GPIO}\left(\frac{R_{LOAD} + R_{analog\ switch}}{R_{Series}}\right) = 5V\left(\frac{351 + 5.5}{367}\right) = 4.857\ V$$

Therefore, the analog switch and MUX cause a voltage drop of 0.143V before reaching the ADC. In order to account for this, when the IC is first received a voltage of 5V will be supplied as in the configuration shown previously. The voltage measurement of the ADC will be compared against a DMM measurement to be within the specifications of the device.

The 390Ω resistor can also be measured using a DMM for a more accurate calculation. This measured value can then be used to calculate each GPIO pin's internal resistance value using the configuration shown in **Figure 7.6**.

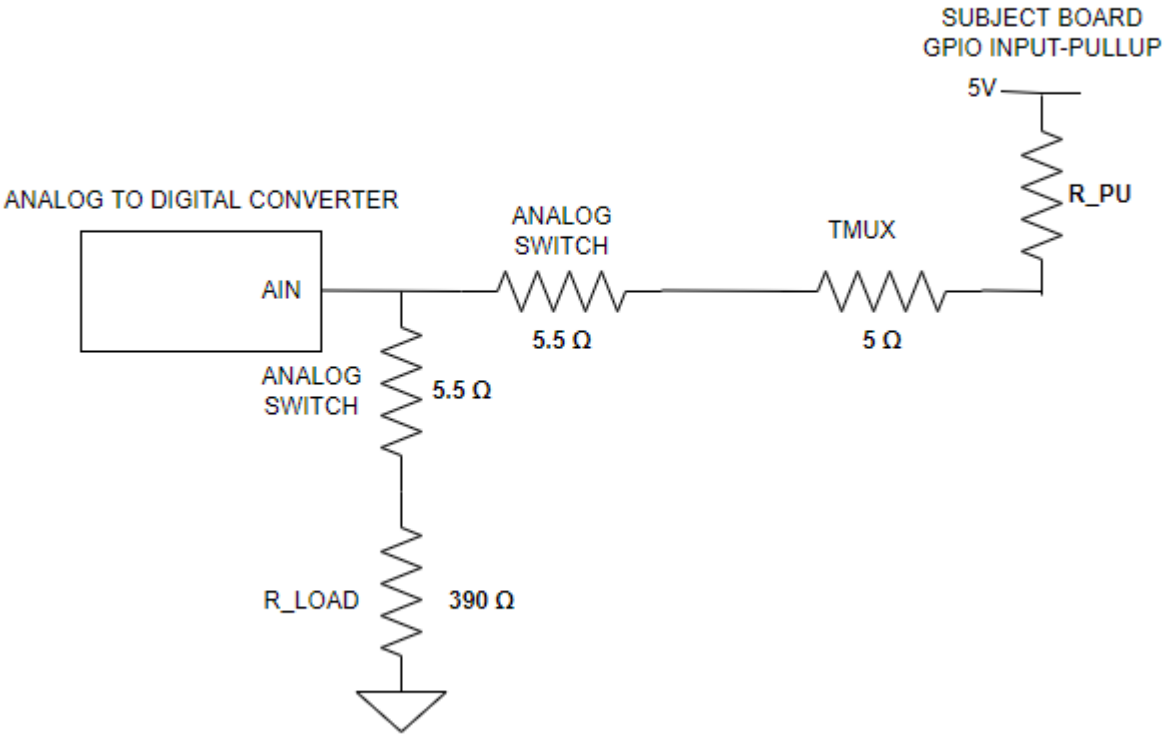


Figure 7.6: GPIO Internal Resistance Measurement Configuration

With the pin configured as input-pullup, the GPIO internal resistance is calculated using the voltage divider equation. The manufacturer states that the internal resistance should be in the range of 20-50 kΩ. The switches and MUX in series with the pull-up resistor have an on resistance that will once again cause a voltage drop before reaching the ADC's input channel. Assuming the maximum on resistances of the MUX and switches to be 9Ω and 10Ω, we have that:

$$V_{AIN} = V_{GPIO}\left(\frac{R_{LOAD} + R_{analog\ switch}}{R_{series} + R_{PU}}\right) = 5\left(\frac{390 + 10}{419 + R_{PU}}\right)$$

$$Let\ R_{PU} = 20k\Omega: \quad V_{AIN} = 5\left(\frac{390+10}{419+20000}\right) = 0.097948\ V$$

$$Let\ R_{PU} = 50k\Omega: \quad V_{AIN} = 5\left(\frac{390+10}{419+50000}\right) = 0.039667\ V$$

Thus, voltage measurements within the range 0.03967 - $0.09795\ V$ corresponds to internal resistance of 20-50kΩ. The pull-up resistance is calculated from the ADC measurement as:

$$V_{AIN} = V_{GPIO}\left(\frac{R_{LOAD} + R_{analog\ switch}}{R_{series} + R_{PU}}\right)$$

$$R_{PU} = \frac{V_{GPIO}\left(R_{LOAD} + R_{analog\ switch}\right)}{V_{AIN}} - R_{series}$$

$$R_{PU} = \frac{5(390 + 10)}{V_{AIN}} - 419$$

When the ADC is not being used for a test, switch U2 is opened such that the ADC is disconnected from the rest of the circuit.

7.3 DAC Circuit

Table 4 shows the requirements and specifications that are met by the DAC.

Table 7: Voltage Generation Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
3.06	The device must test and record the GPIO **input pin logic thresholds** of the subject board	3.06.1	The device must test that logic low/high is produced when given a voltage within their threshold
3.07	The device must test and record the **accuracy** of the subject board's **ADC pins and channels**	3.07.1	The device must generate voltages for the subject board's ADC pins
		3.07.2	The device must test the measured voltages from the ADC pins
3.08	The device must test and record **all of the pins** capable of **GPIO** wakeups from **sleep mode**	3.08.1	The device must generate a signal to wake the subject board
		3.08.2	The device must test that the subject board's GPIO pins

			can exit sleep mode

The DAC shown in **Figure 7.7** is the DAC121C081, a 12 bit digital to analog converter with I²C compatibility that operates using the Pi's 5V supply as a voltage reference.

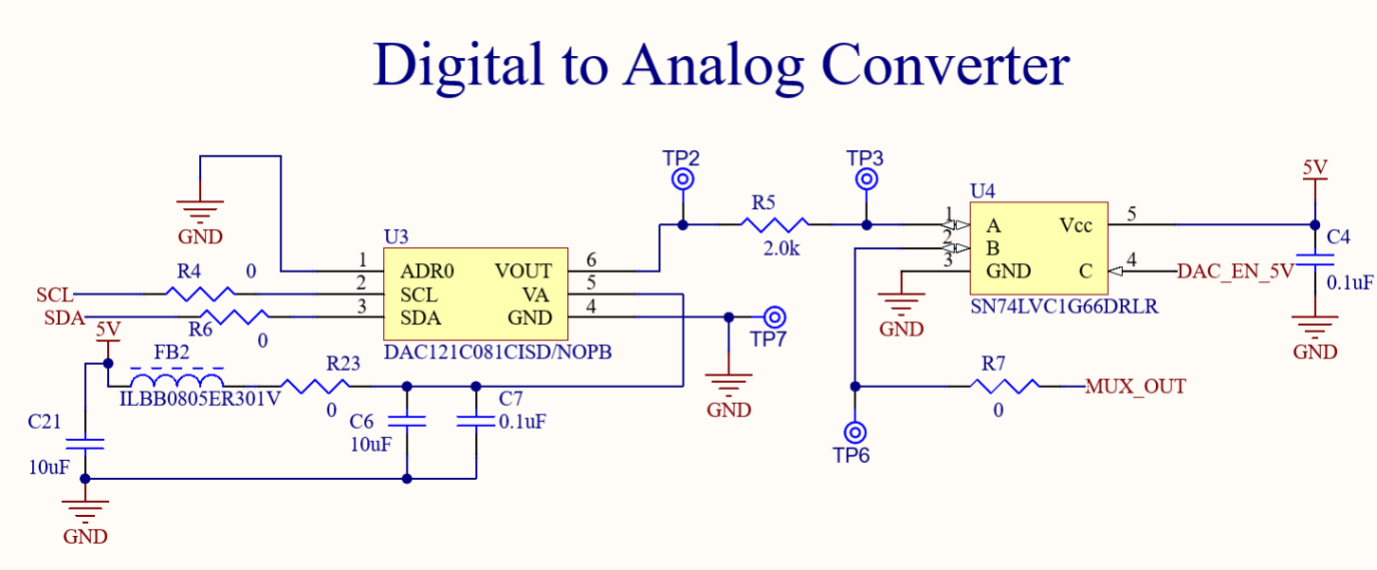


Figure 7.7: Digital to Analog Converter Integrated Circuit

Its single channel output can generate voltages from 0-4.989V. The resolution of the DAC is calculated as:

$$Resolution = \frac{V_{REF}}{2^N} = \frac{5V}{2^{12}} = 1.22\,mV$$

The 2 kΩ resistor R1 is placed in series with the DAC to limit the output current in the event that the subject board's pin is shorted. This also serves as protection in the case that the DAC is accidentally used to sink current. The two capacitors on the voltage supply line to the DAC are used to eliminate any high frequency noise and supply the DAC with extra bursts of current as needed when making conversions, as recommended by the datasheet. A ferrite bead was also placed in series to further eliminate unwanted high frequency noise.

The DAC is programmed by the Pi over the SDA line. The Pi sends the DAC a digital code corresponding to a voltage used for testing the subject board's input pins. These voltages are used in testing the subject board's logic level thresholds, ADC, and GPIO wakeup from sleep mode.

Logic level thresholds are tested by generating voltages just above or below the manufacturer's rated thresholds. The input logic levels for the arduino are 0-1.5V for logic low (V_{IL}) and 3-5V for logic high (V_{IH}). For STM the input logic levels are 0-1V for logic low (V_{IL}) and 2.31-3.3V for logic high (V_{IH}). If a voltage generated within the manufacturer's rated threshold is not detected by the subject board, then the pin is determined to be faulty.

The subject board's ADC is tested by generating a set of voltages across the input range of the ADC, as seen in **Figure 7.8**.

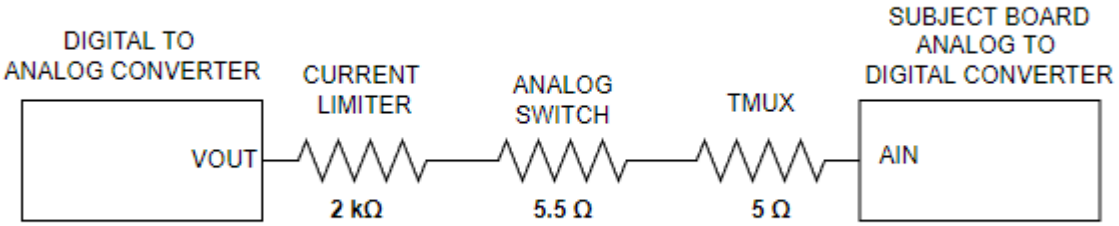


Figure 7.8: Subject Board ADC Test Configuration

The internal resistance of the Arduino's ADC is known to be about 100MΩ. The voltage drop across the resistor, MUX, and switch becomes negligible, as shown below.

$$Let\ V_{out} = 5V:$$

$$V_{ADC} = V_{out}\left(\frac{R_{ADC}}{R_{ADC}+R_{current\ limit} + R_{MUX} + R_{Analog\ switch}}\right)$$
$$= 5\left(\frac{100{,}000{,}000}{100{,}000{,}000 + 1000 + 5 + 5.5}\right) = 4.99995$$

Therefore, even at the full scale range of the DAC, the effect of the resistor, MUX, and analog switch is only a 0.00005V drop and can be ignored. Voltages from 0 to 5V will be generated in 0.5V increments to test the subject board's ADC. The subject board's voltage measurement will be reported to the Pi via USB serial communication and compared against what was sent to it to be within an acceptable range.

Lastly, the DAC is used to wake the subject board from sleep modes. Pins 2 and 3 on the arduino can be configured as interrupt pins, which when triggered by an external voltage source can wake the subject board from sleep. By monitoring the current draw of the arduino (explained in the following section) it is known whether or not the subject board has exited sleep mode. When the DAC is not being used for a test, the switch U4 is opened, disconnecting the DAC from the rest of the circuit.

7.4 Current Measurement

Table 5 shows the requirements and specifications that are met by the current measurement IC.

Table 8: Current Measurement Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
2.06	The device must **accurately measure** the subject board **currents** under test	2.06.1	The device must implement hardware that measures current with an accuracy of 10 uA within the range 0 to 1000uA (subject to change based on IC)
		2.06.2	The device must implement hardware that measures current with an accuracy of 10 mA within the range 0 to 1 A (subject to change based on IC)
3.08	The device must test and record **all of the pins** capable of **GPIO** wakeups from **sleep mode**	3.08.1	The device must generate a signal to wake the subject board
		3.08.2	The device must test that the subject board's GPIO pins can exit sleep mode
3.09	The device must test all **sleep modes and power modes**	3.09.1	The device test the subject board's current draw during sleep and power modes to be below a configurable threshold

The current measurement IC shown in **Figure 7.9** is the INA219BID, a current monitor that measures the small voltage drop across a shunt resistor by using a 12 bit ADC. The measured voltage drop is multiplied with the shunt resistance value to calculate the current flowing through the resistor.

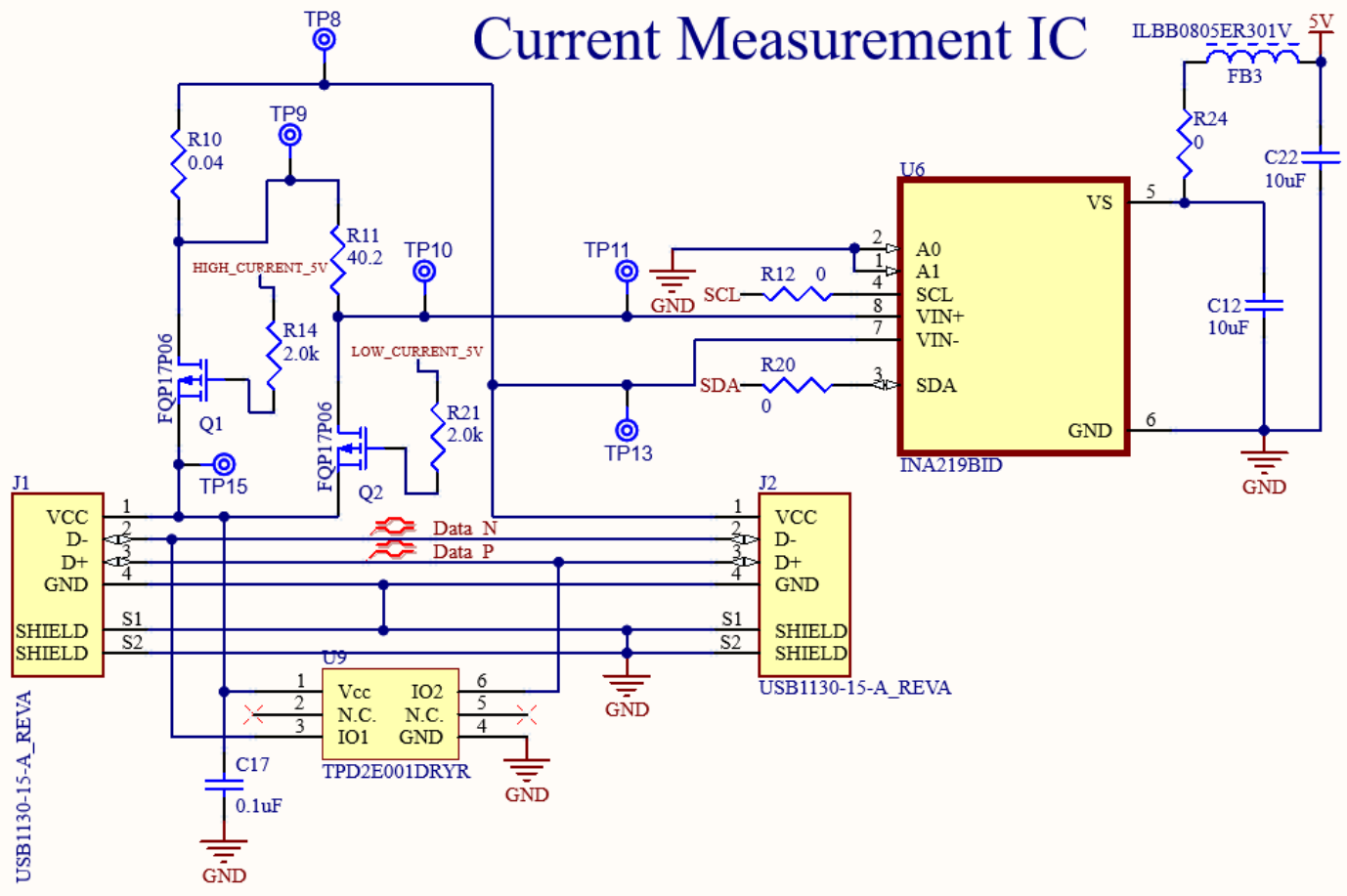


Figure 7.9: Current Measurement Integrated Circuit

The full scale range of the IC is set to 40mV, which corresponds to a resolution of:

$$Resolution = \frac{Full\ Scale\ Range}{2^N} = \frac{40\ mV}{2^{12}} = 9.766 \simeq 10\mu V$$

Current measurements are used in testing the subject board's current draw during normal operation and in sleep mode. J1 and J2 seen in **Figure 7.9** are USB Type A connectors. The input USB J1 is fed from the USB hub located within the device, while the output USB J2 will lead to the subject board. By routing the USB current through the PCB, measurement of the USB supply current is possible.

The filter consisting of R6, R12, and C13 is used to eliminate any high frequency noise and ensure a clean measurement, as recommended by the datasheet. It was found, however, that this filter can cause an input bias to the IC. It was determined that the filter is not necessary for the board since high frequency noise should not be present on the voltage supply, and was removed from the final implementation.

Two separate configurations are possible from the circuit seen in **Figure 7.9**. In one configuration, the transistor Q1 is closed and Q2 is opened such that current flows through the 40 mΩ resistor R3. This simplified configuration is seen in **Figure 7.10**.

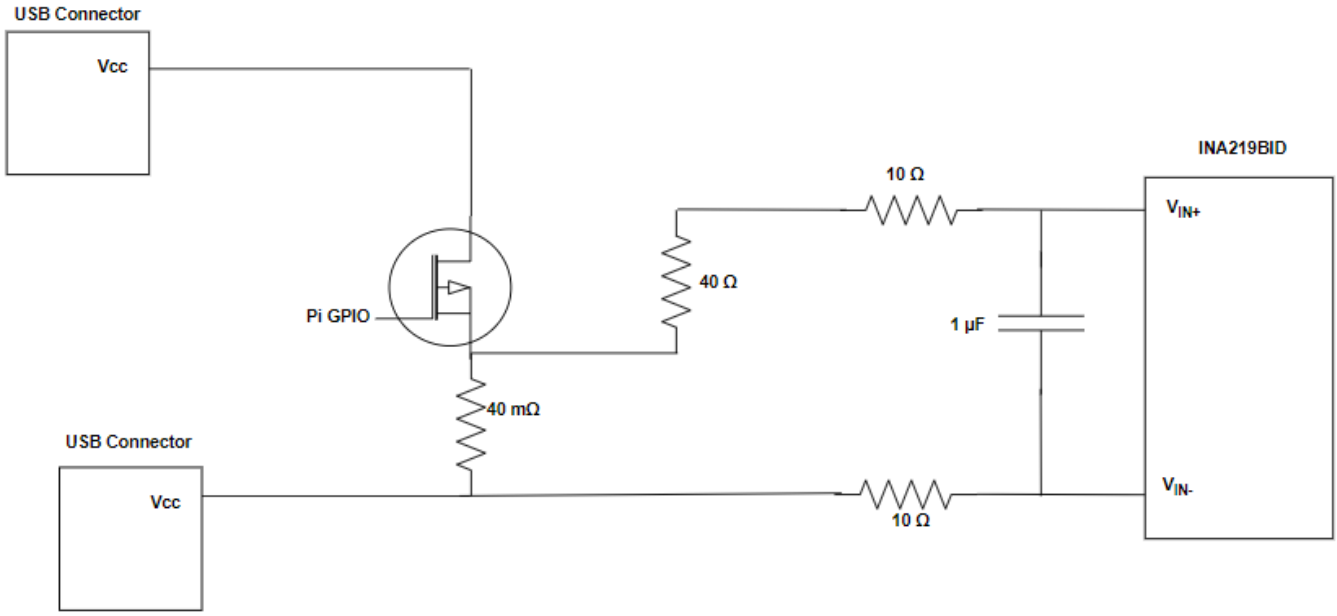


Figure 7.10: Current Measurement Configuration for 0-1A with 10mA accuracy

Current is redirected through the 40mΩ shunt resistor in order to generate a voltage drop for the INA219BID current sensor to measure. This configuration is used to measure currents from 0 - 1A with an accuracy of 10mA. The input channels of the IC are assumed to have very large input impedance and act as an open circuit, so the 40.2Ω resistor connected to VIN+ will have no effect on the voltage reading since no current is flowing through the

resistor and there is no voltage drop across the 40.2Ω resistor. The voltage drop across the 40 mΩ shunt resistor corresponding to the minimum and maximum currents to be detected in this configuration are calculated as follows:

$$V_{IN+} - V_{IN-} = R_{shunt} * i_{USB}$$

$$Let\ i_{USB} = 10\ mA: \quad V_{IN+} - V_{IN-} = 40m\Omega * 10mA = 400\mu V$$

$$Let\ i_{USB} = 1\ A: \quad V_{IN+} - V_{IN-} = 40m\Omega * 1A = 40mV$$

Therefore, 10mA is detectable by the IC as about 40 LSBs of the ADC. Currents up to 1A are also measureable, which corresponds to the full scale voltage drop allowed by the IC. These calculations were confirmed in LTSpice and can be found in **Appendix E**.

For measuring currents from 0-1mA, the alternative configuration is used where transistor Q2 is closed and Q1 is opened, as seen in the simplified circuit of **Figure 7.11**:

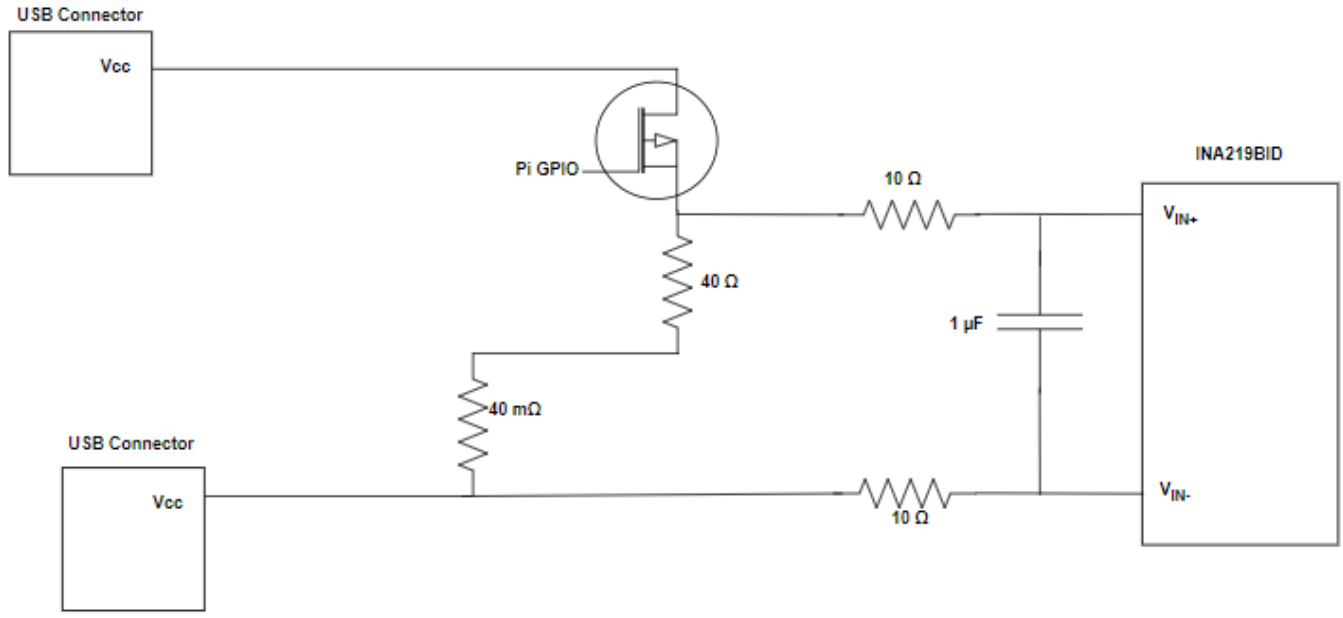


Figure 7.11: Current Measurement Configuration for 0-1 mA with 10µA accuracy

In this configuration, current is redirected through the 40.2Ω and 40mΩ resistor, which form a series shunt resistance of 40.24Ω. This configuration is used to measure current from 0-1mA with an accuracy of 10µA. The voltage drop across the shunt resistor corresponding to the minimum and maximum currents to be detected in this configuration are calculated as follows:

$$V_{IN+} - V_{IN-} = R_{shunt} * i_{USB}$$

$$Let\ i_{USB} = 10\ \mu A: \qquad V_{IN+} - V_{IN-} = 40.24\ \Omega * 10\mu A = 402.4\mu V$$

$$Let\ i_{USB} = 1\ mA: \qquad V_{IN+} - V_{IN-} = 40.04\ \Omega * 1mA = 40.24mV$$

Therefore, 10µA is detectable by the IC as about 40 LSBs of the ADC, as well as currents up to 1mA which corresponds to the full scale range voltage of the IC.

In either of the current measuring configurations, the on resistance of the transistor Q1 or Q2 will not have an effect on the current measurement since the transistors are connected in series, and the subject board will draw current as needed.

By monitoring the USB supply current during and after sleep modes, the device can confirm that the subject board is capable of GPIO wakeup from sleep mode. It can also monitor that the subject board is not drawing more current than it should. The IC is operated by the Raspberry Pi using the SDA line. In order for the IC to make current calculations, the Pi must program the Calibration register. The Calibration register sets the current that corresponds to the full-scale voltage drop across the shunt resistor and is calculated according to the following equation:

$$Calibration\ Register\ =\ trunc\left[\frac{0.04096}{Current\ LSB\ *\ R_{Shunt}}\right]$$

Where the *Current LSB* is the programmed value for the least significant bit of the current register and is calculated as:

$$Current\ LSB\ =\ \frac{Maximum\ Expected\ Current}{2^{15}} = \frac{1\ A}{2^{15}} = 3.052\ \mu A$$

$$\Rightarrow Calibration\ Register\ =\ trunc\left[\frac{0.04096}{3.052\ \mu A\ *\ 40m\Omega}\right] = 33554$$

Note that the above shunt resistance chosen was 40mΩ, which corresponds to the configuration for measuring 0-1A. With the calibration register configured, the IC is ready to calculate current from its voltage measurement. The value in the current register is determined by the following equation:

$$Current\ Register\ = \frac{Shunt\ Voltage\ Register\ x\ Calibration\ Register}{4096}$$

where the shunt voltage register is determined by:

$$Shunt\ Voltage\ Register\ = \frac{Shunt\ Voltage}{Shunt\ Voltage\ LSB}$$

Assuming maximum shunt voltage of 40 mV (corresponding to 1A of current), the current register equates to:

$$Shunt\ Voltage\ Register\ = \frac{40\ mV}{10\ \mu V}\ =\ 4000$$

$$Current\ Register\ = \frac{4000\ *\ 33554}{4096} = 32767.578$$

Finally, the current is calculated as:

$$Current\ =\ Current\ Register\ *\ Current\ LSB\ = 32767.578\ * 3.052\ \mu A\ = 0.999987\ A$$

Which is very close to the hand calculation of 1A.

For measuring smaller currents from 0-1mA, the device can be recalibrated using the series 40.24mΩ shunt resistance. Using this alternative shunt resistor value and a maximum expected current of 1mA the IC calculations are redone as:

$$Current\ LSB\ = \frac{Maximum\ Expected\ Current}{2^{15}} = \frac{1\ mA}{2^{15}} = 30.517\ nA$$

$$Calibration\ Register\ =\ trunc\left[\frac{0.04096}{Current\ LSB\ *\ R_{Shunt}}\right] =\ trunc\left[\frac{0.04096}{30.517nA\ *\ 40.24\Omega}\right]$$
$$= 33354$$

$$Shunt\ Voltage\ Register\ = \frac{Shunt\ Voltage}{Shunt\ Voltage\ LSB} = \frac{10\mu V}{10\mu V} = 1$$

$$Current\ Register\ = \frac{Shunt\ Voltage\ Register\ x\ Calibration\ Register}{4096} = \frac{1\ *\ 33354}{4096}$$
$$= 8.143$$

$$Current = Current\ Register\ *\ Current\ LSB = 8.143\ *30.517\ nA\ = 0.2485\ \mu A$$

Therefore, the IC can measure current with a resolution of 0.2485 µA. The registers on the IC are volatile and the Calibration register must be reprogrammed every time the IC reboots. In testing of the subject board's sleep modes, it was discovered that current draw in all sleep modes never drops below 10mA. For this reason, the low current measurement configuration is not necessary and can be removed from the circuit if desired.

7.5 Circuit Protection

Table 6 shows the requirements and specifications for circuit protection of the device.

Table 9: Circuit Protection Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
2.02	The device must implement **circuit protection to avoid damage** to its hardware	2.02.1	The device must utilize overvoltage / surge protection IC's for circuits that have potential to experience overvoltage

The device's custom circuit implements transient voltage suppression (TVS) diodes in order to protect components that the user directly interacts with. The user will be responsible for plugging the USB cable into their subject board. The TPD2E001 IC is used on the USB channels to protect from overvoltage. The IC is a two channel TVS with very low capacitance, making it ideal for high speed signals. The central diode seen in **Figure 7.12** acts as the ESD clamping diode, while the 2 hiding diodes per line are used to reduce capacitive loading.

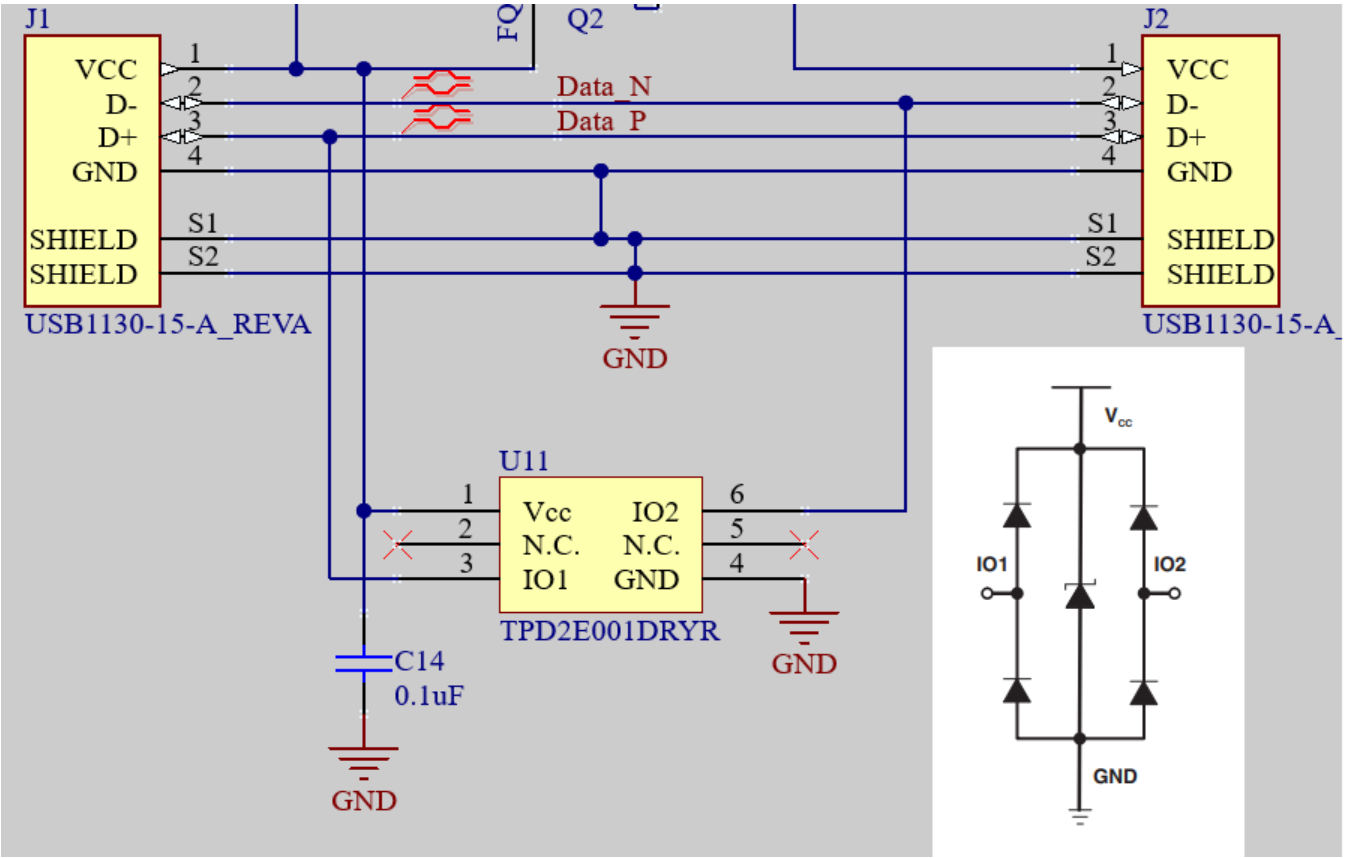


Figure 7.12: Implementation of the TPD2E001 TVS

The IC's intended application is to protect all of the pins on a USB 2.0 connector. It is rated for ±8000V under standard IEC 61000-4-2 (contact discharge) and ±15000V using the human body model. The channel input capacitance is 1.5pF. A bypass capacitor is placed from VCC to ground as recommended by the manufacturer.

One other area that the user interacts with is when they connect the subject board directly to the device's header pins. The header pins are connected to the MUXs which do not have sufficient overvoltage protection to protect from ESD. For this reason, the D3V3F8U9LP3810-7, an 8 channel TVS diode, was used for each of the signals

connecting to a MUX on LittleFoot. In addition to TVS diodes, a ground plane exists on the device for the user to touch before connecting their subject board, which can be used to prevent potential shocks to the board.

The circuit also implements isolation for the ICs and pins on the subject board. The DAC and ADC each have a SN74LVC1G66 bidirectional switch in series with the input/output channels on the IC. If the DAC or ADC is not being used for a test, the switch is opened such that the IC is disconnected and isolated from the rest of the circuit.

The multiplexers also act as isolation for each of the pins on the subject board. Having only one multiplexer enabled at any one time and with proper control of the switches, it can be guaranteed that only one input and output are connected to the subject board at a time.

7.6 PCB Design

The BigFoot and LittleFoot PCB layouts were designed in Altium as 4 layer boards. The first top layer is a signal layer. This signal layer contains the bulk of the potentially sensitive signal line traces and component mounting pads. The second layer is a ground plane that serves three main purposes. First, the ground plane layer will act as a shield for electromagnetic interference. Second, the ground plane allows for shorter return paths to ground, which aids in minimizing the length of potential ground loops and minimizing return-path resistances. Third, the ground plane adds a minor amount of decoupling capacitance. When all of these factors are accounted for, the ground plane layer will effectively improve signal integrity as a whole.

The third layer is a power plane for 5V power. The power plane's implementation results in two main benefits. First, it aids in reducing the lengths of traces used to provide various components with 5V power. Second, its placement below the ground plane allows it to mimic the behavior of a bypass capacitor; thus, it will slightly improve the regulation of 5V power wherever needed. Considering these two factors, the power plane serves to provide cleaner 5V power where needed while also reducing the noise it creates when supplying power. Lastly, the bottom signal layer contains the traces and component mounting pads that are used to route power and DC digital signals. This layer serves to both reduce board size and keep signals that are prone to emit noise isolated from critical low integrity signals.

Stitching vias are placed on both the top and bottom signal layers, along with fencing vias being used to surround critical signal traces such as the MUX_OUT signal used for testing the subject board's pins. The stitching vias aid in providing a shorter return-path to the ground plane, mitigation of ground loops and their lengths, and improved immunity of EMI noise. The fencing aids in further isolating critical low integrity signal traces and component mounting pads.

The PCB layout for the BigFoot board can be seen below in Figure 7.13.

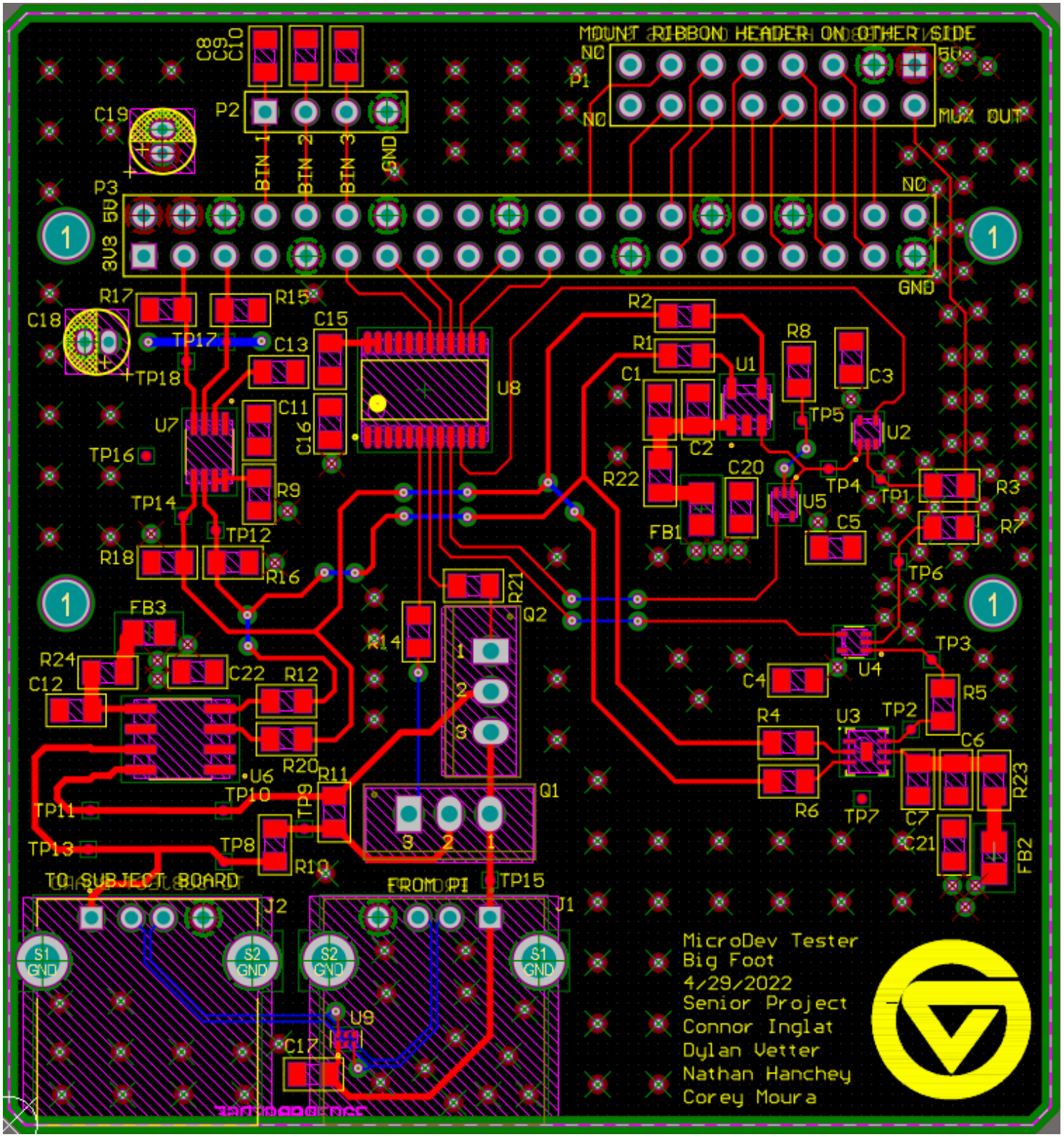


Figure 7.13: BigFoot PCB Layout

There were several main considerations when doing the BigFoot PCB layout. The main challenge with any PCB layout is part placement. It was desired that each measuring component of the PCB was to be separated not only by the analog switches connecting them, but in physical space as well. This was done to keep the layout simplified, avoid crosstalk between different parts of the PCB, and to aid in the debugging process when it came time to work on individual parts

of the circuit. One of the harder traces to route were the SDA and SCL line, which had to be connected to the current measurement IC, the ADC, and the DAC. Since these traces span mostly the length of the entire board, they had to be routed in a clever way, sometimes using vias to travel onto the bottom layer in order to cross over other traces on the board.

The placement of the USB connectors also had to be taken into consideration with the enclosure in mind. These connectors needed to be placed on the edge of the board and towards the bottom of the PCB to allow for connection of the USB cable external to the enclosure. The most efficient way to route the USB traces was to place one USB connector on the top of the board and the other on the bottom. This allowed the data lines to be connected in a mostly direct path, while also touching off on the connected TVS diode. With routing the rest of the traces on the board, the goal was to keep the trace lengths as short as possible and to avoid acute and right angles. The trace of most concern was the MUX_OUT signal trace, which is connected to LittleFoot for testing of individual pins on the subject board. This trace was surrounded by fencing vias in order to protect from noise. The ADC and DAC circuits were placed as nearest as possible to the MUX_OUT trace while keeping those circuits spaced as far as possible.

All other traces on the board were simple to route due to the initial planning of part placement. Polygon pours with vias inside were made for all of the power connections to individual components. Lastly, stitching vias were placed throughout the board. With everything set in place, the silkscreen layer containing component designators and other useful information was tidied up so that designators were oriented in the same directions. This helped in avoiding any ambiguity and further aided in assembly and debugging of the board.

The boards were manufactured by PCBWay and were shipped from China with a stencil. The stencil was used to populate the surface mounted components of the board. After populating, the board was placed in an oven for each of the components to be set into place. Through hole components were then soldered by hand onto the BigFoot PCB. One unexpected issue with the BigFoot PCB population was the analog switches, which didn’t always set into place properly. It was discovered in testing that this caused some signals to be grounded undesirably. This was resolved by simply hand soldering those components using a microscope to set them into place perfectly. For one final touch, hot glue was used to separate the leads of the electrolytic capacitors since they were prone to touching and being shorted together. This completed the BigFoot PCB assembly.

For the LittleFoot PCB layout, much of the same considerations for BigFoot were also copied. The PCB Layout can be seen in Figure 7.14.

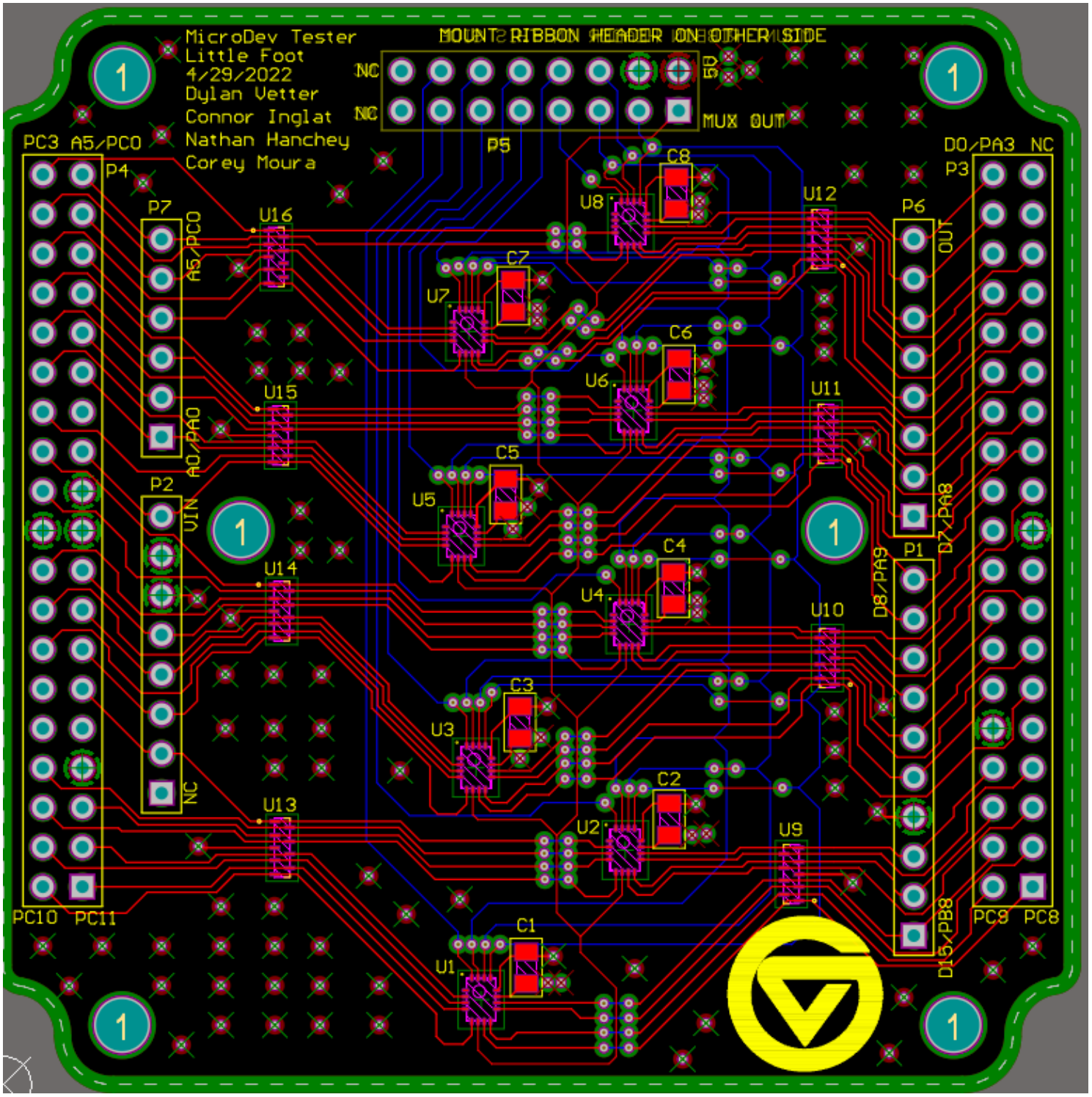


Figure 7.14: BigFoot PCB Layout

The most challenging part for LittleFoot was once again the part placement, which determines how the entire board traces are done. The placement of the header pin interface was predetermined by the location of pins on the subject boards. To get these header pins and mounting holes perfectly into place, a .dwg file was used on the board overlay to align the holes and then the header pins were locked into place. Each pin of the subject board needed to be routed to an individual pin on one of the multiplexers. With that idea in mind, MUXs were staggered in a vertical pattern as seen in Figure 7.14 so that traces from each side of the board could be routed to a MUX. The TVS diodes also needed to be taken into consideration, since every pin trace must travel through a TVS diode for ESD protection. These were placed

in a way to allow easy routing to the MUXs.

The enable and address pins for the MUXs were routed on the bottom layer of the board since the pin traces were on the top. This way, enable traces could be directly routed to each MUX without having to use more than one via. Since the address traces inevitably needed to cross at certain points, routing them required some creativity in staggering their positions and using clever vias to route around each other, as seen running down the bottom layer of the right side of the board. Since each MUX is oriented in the same way, the pattern could be repeated for the entire length of the board. Lastly the MUX_OUT signal for each MUX was routed down the center of the board into the ribbon cable pin connecting them to BigFoot. Stitching vias were placed throughout the board as was done on BigFoot.

The LittleFoot PCBs were ordered from PCBWay and shipped from China with a stencil. The stencil was used to smear solder paste onto the board so that components could be placed and the board could be baked in an oven. Once the boards were baked, the biggest challenge was adjusting the MUXs to have proper connection. Since a pick-and-place machine was not used to place the components and because the MUXs were so small, the parts didn't settle perfectly into place. Each MUX had to be hand soldered, which proved to be extremely difficult. The width of the pins on the MUX were bigger than the tip of the soldering iron, so getting solder to adhere to just one pin at a time was a challenge. Having to do this for 8 MUXs across 10 boards proved to be a mistake in the design. If the project were to be redone, this is the main thing that should be changed. Once all MUXs were hand soldered onto the boards, through hole components were added and the assembly was finished. Upon validating LittleFoot boards, it was discovered that the TVS diodes were failing and causing tests to fail. It is unknown whether these were bad parts or if the damage was caused in development of the testing program. In a new revision, it might be beneficial to remove the TVS diodes which were done for some of the LittleFoot boards.

7.7 Hardware Validation

The equipment used to validate the hardware of this device are listed in the table below.

Table 10. Validation tools

Specification	Tool
Equipment	Keithley 2110 5½ Digit Multimeter
	Keithley 2231A-30-3 Triple Channel DC Power Supply
	Tektronix MSO 2022B Mixed Signal Oscilloscope
Software	LTSpice v17.0.34

The hardware functionality is verified via measurement or inspection for every specification. These validation methods are outlined in the table below for hardware tests requiring measurements. A full list of the validation testing and results per board can be found in Appendix P.

Table 11. Hardware validation methods

Specification #	Validation Method	NOTES
ADC TESTS		
2.05.1	The ADC must measure voltage with an accuracy of 0.1V. Using a separate power supply, 5V will be applied to the ADCs input channel. Using a DMM, the voltage at the ADC's output terminal will be measured. This measured value will be compared against the ADC's measured value to be within 0.1V of the DMM's value.	**DMM Measurement:** TP4 to GND
INA TESTS		
2.06.2	The current sensor must measure current with an accuracy of 10mA from 0-1A. Loads of 5, 10, 20, 50, and 500 Ω will be connected through the USB terminal. Current out of this terminal will be measured with a DMM and compared against BigFoot's measurement. BigFoot's measured value must be within 10mA.	INA219 Removed filter from boards: 2 / 6 / 10
DAC TESTS		
3.06.1	The device will configure the DAC to output several voltages within the range of logic low and logic high of the subject board. A DMM will be used to measure the output voltage of the DAC. The measured voltage will be compared against the voltage code the DAC was provided.	DMM Measurement: TP2 (Vout) to TP7 (GND)

3.07.1	The device will configure the DAC to output several voltages within a range for testing the subject board's ADC. A DMM will be used to measure the MUX out voltage. The measured voltage will be compared against the mux voltage code the DAC was provided. (Test ensures analog switches are working correctly)	DMM Measurement: TP(TP7) to (On MUXout pin)
MUX TESTS		
N/A	LittleFoot will be controlled using a separate power supply. Enable and Address pins will be supplied 5V in combinations that will test each MUX pin at least once. When a pin is enabled, it will be supplied 3.3V. This voltage will be checked using a DMM on the MUX output signal to be close to the 3.3V provided.	
INTEGRATION TESTING		
N/A	Using an STM board without any faults, BigFoots and LittleFoots are placed in pairs used to test the working STM board. The test is expected to pass in all categories, proving that the BigFoot and LittleFoot Pair are working properly.	

Software Design

The software design consists of two programs used to run the device and the subject board during the testing of the subject board. First, the device's program is written in a combination of Python and Command Line Interface script. This software will allow the device to run testing on the subject board it's connected to; it will also be used to implement a graphical user interface (GUI) between the user and the device, which will allow for a user to fluidly interact with the device and run tests on a desired subject board.

Second, the subject board's program is written in embedded C/C++, and it will be flashed onto the subject board during testing of the subject board in question using the device program's bash script. This software will allow the device to interact with the subject board being tested. Specifically, it will facilitate the ability for the device to communicate with the subject board via serial communication. Additionally, it will allow the device to manipulate the subject board as required for a portion of its testing. All the software can be accessed through a github directory at the url *https://github.com/hancheyn/MicroDev*.

Important Terminology:

Subject Board - This refers to the development board under test.

Pin Mapping - This refers to the method a value given to a pin will be linked to a multiplexer address in which the pin is connected.

The figure below illustrates an overview of the software. The device's software, which will run from a Raspberry Pi Zero 2W, located within the device itself, will be the central hub for all hardware peripherals used within the device. This is in **Figure 8.0**.

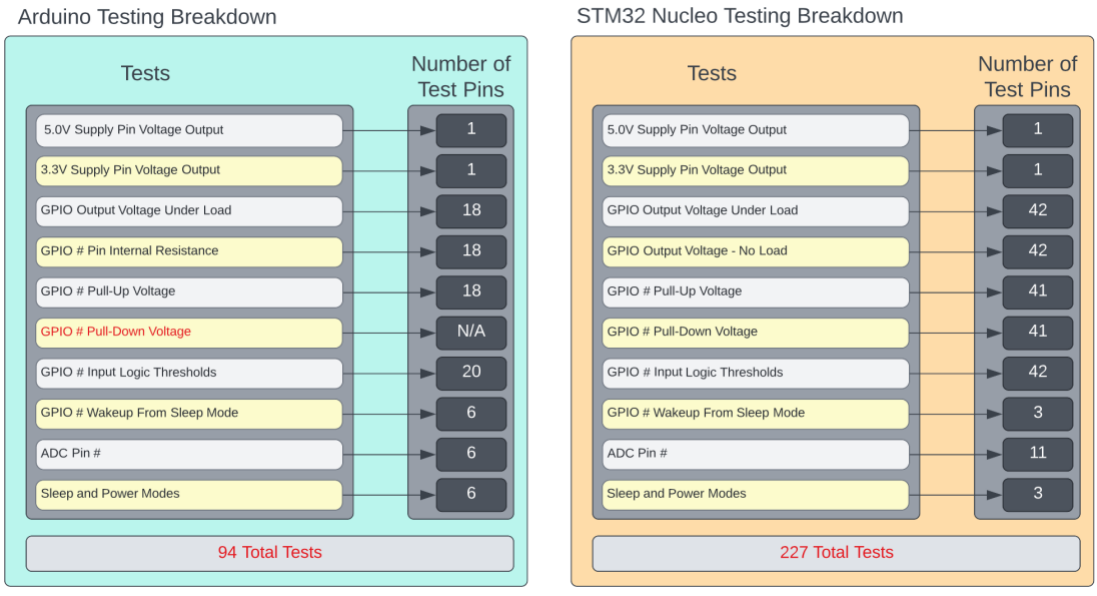


Figure 8.0: Software Test Overview

8.0 Software Requirements and Specifications

The requirements and specifications depicted in **Table 7** below are all required, and they will be satisfied by both the subject board's and device's programs.

Table 12: Software Requirements and Specifications

REQ #	REQUIREMENT	SPEC #	SPECIFICATION
4.00	The device must **identify the subject board being tested**	4.00.1	The device's software must read and record the subject board's identification information found in the Device Descriptor Table
4.01	The GUI implemented on the device must be **intuitive to use**	4.01.1	Any interactive icons on the device's GUI must be labeled
4.02	The device must allow the user to **start** a **test through** its **GUI**	4.02.1	The device's GUI must display a "begin test" button once the subject board is properly loaded
4.03	The device's GUI must **provide a detailed report** of the subject board's test	4.03.1	The device's GUI must display a report of the failures of the subject board at the conclusion of testing
		4.03.2	The device must report if an error occurred in the execution of a test
4.04	The device's software must **allow configurable thresholds for tests**	4.04.1	The device's software must use removable media to load a configuration file
4.05	The software developed for the device must be **appropriately documented**	4.05.1	The device's software must be commented
		4.05.2	The device's software must be documented with a manual
3.00	The device must test and record the **voltage supplied** from the subject board's **5V and 3.3V output pins**	3.00.1	The device must test that the supply voltage is within the configurable range
3.02	The device must test and record each GPIO pin's output voltage under **load sourcing**	3.02.1	The device must test that the pin's output voltage remains above a configurable threshold
3.03	The device must test and record each GPIO pin's internal resistor value	3.03.1	The device must test that the calculated internal resistance of each GPIO pin is within the manufacturers range
3.04	The device must test and record each GPIO pin's internally resistive **pull-up voltage while unloaded**	3.04.1	The device must measure voltage from each GPIO to GND while pin is configured as pull-up
		3.04.2	The device must measure and test each GPIO pin's voltage while configured as pull-up to be within a configurable range
3.05	The device must test and record each GPIO pin's internally resistive **pull-down voltage while unloaded**	3.05.1	The device must measure and test each GPIO pin's voltage while configured as pull-down to be within a configurable range
3.06	The device must test and record the GPIO **input pin logic thresholds** of the subject board	3.06.1	The device must test that logic low/high is produced when given a voltage within their threshold
3.07	The device must test and record the **accuracy** of the subject board's **ADC pins and channels**	3.07.1	The device must generate voltages for the subject board's ADC pins
		3.07.2	The device must test the measured voltages from the ADC pins
3.08	The device must test and record that subject is capable of **GPIO** wakeup from **sleep mode**	3.08.2	The device must test that the subject board's GPIO pins can exit sleep mode

9.0 Description of The Device's Program

The device's program implements the model-controller-view (MCV) design pattern in Python, which is used to both facilitate testing of the subject board and manage the GUI of the device. The device's program additionally utilizes the **serial** Python library to handle serial communication between the device and subject board. The **subprocess** Python library will be used for executing terminal processes and communicating through pipes to the Python process. Additionally, the **time** library will be used for delays between certain operations. An example of the MicroDev setup procedure, which includes library setup in setup.sh, can be found in **Appendix F**. The Command Line Interface developed by the subject board manufacturer is controlled by terminal calls that handle flashing the subject board's program onto it as well before testing begins. It will also be used to compile the subject board test program upon boot up of the Raspberry Pi Zero. These relationships are shown below in **Figure 9.0**.

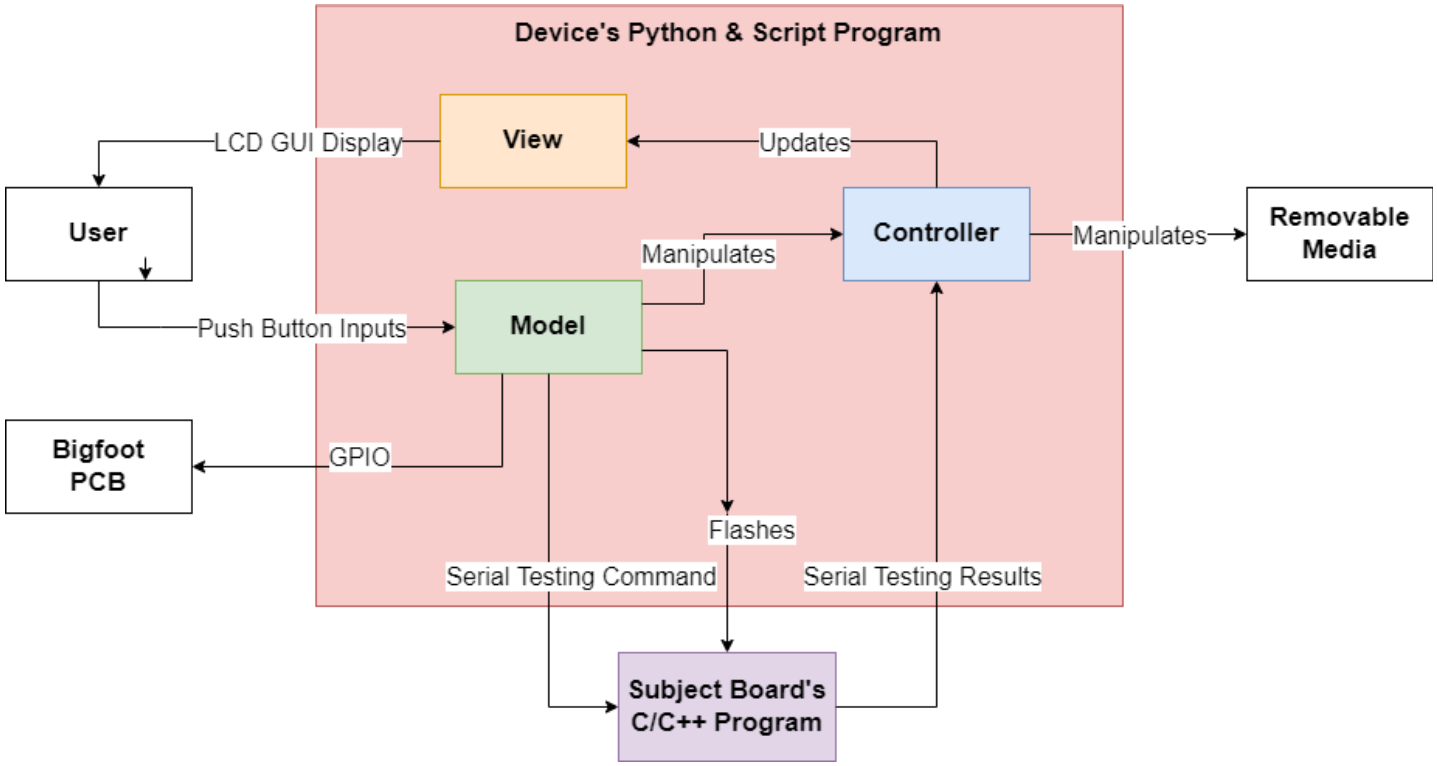


Figure 9.0: Device Software Design Overview

Setup for the Raspberry Pi includes the installation of software for communication protocol and command line interfacing. The Arduino CLI or *Command Line Interface* (arduino-cli) and the STM CLI (st-link) is uploaded for identification of the subject board and flashing the test program to the subject board. Following setup and installation of the MicroDev program, the Raspberry Pi Zero Operating System will be reconfigurable as read only to mitigate corruption of the SD card memory while writing is not taking place. The Raspberry Pi program utilizes configuration and output files. In the event that the SD card needs to be replaced a new Micro SD card will need to be purchased. The Raspian OS can be found in this link: *https://www.raspberrypi.com/software/.* Additionally, the documentation shall include a copy of the software to upload onto a new SD card and a procedure for the setup, described in **Appendix F**.

The threshold configuration file for a given subject board is named with the following convention, *[subject]Threshold.config*. The preconfigured STM32 Nucleo file is named *stm32f4Threshold.config* and the Arduino

Uno is named *unoThreshold.config*. The threshold file is setup as follows in Figure 9.0.1:

```
1   TestID,ThresholdValue1,ThresholdValue2,...
2   1,3.5,0.5
3   2,3.5,0.5
4   3,20000,50000,4.0
5   4,20000,50000,0.5
6   5
7   6,5,3.3,1.5,1,0
8   7,6
9   8,6
10  M,4,2.5
```

Figure 9.0.1: Threshold Configuration Example

After the first line, each following line is the test thresholds that will be used as the pass or fail conditions of the test. The first line describes the threshold configuration format, an improperly formatted threshold file will end the testing process. The second line represents Test #1 and each following line must be the next consecutive Test ID. M, or miscellaneous, goes on line 10. The thresholds are as follows:

Test #1 | Min High Voltage | Max Low Voltage
Test #2 | Min High Voltage | Max Low Voltage
Test #3 | Min Resistance | Max Resistance | Pull-Up Min Voltage
Test #4 | Min Resistance | Max Resistance | Pull_Down Max Voltage
Test #5 | N/A
Test #6 | Voltages Generated By DAC (user can add more test voltages)
Test #7 | Current Drop mA (initial - final)
Test #8 | Current Drop mA (final - initial)
Misc. | 5V Pin Minimum Voltage | 3V3 Pin Minimum Voltage

The test configuration file for a given subject board is named with the following convention, *[subject]Test.config*. The preconfigured STM32 Nucleo file is named *stm32f4Test.config* and the Arduino Uno is named *unoTest.config*. The test file is setup as follows in Figure 9.0.2:

```
1   TestID,PinID,MuxAddress,MuxEnable
2   1,21,4,3
3   1,23,6,3
4   1,30,5,4
5   1,32,7,4
6   1,37,4,5
7   1,39,6,5
8   1,40,7,5
9   1,45,4,6
10  1,47,6,6
11  1,53,4,7
12  1,55,6,7
```

Figure 9.0.2: Test Configuration Example

After the first line, each following line is a test that will be conducted. The first line describes the test configuration format. An improperly formatted test configuration file will end the test before finishing. The Test ID and Pin ID that is recognized by the subject development board goes first. Then, the Multiplexer Address and Enable pin comes next, all separated by commas. It is important to remember the pin ID must go with the *address* and *enable value* for that pin. The exception is with tests #7 and #8. Test #7 does not need the address and enable value and Test #8 has a preconfigured wakeup pin depending on the subject board. For the STM sleep mode tests, the STM has to be restarted to reconfigure the original state of the pins before it can be put into low power mode. The restart test is *9,0,3,3* where the number 9 indicates a restart and 3,3 is the multiplexer enable and address.The sleep also does not require a pin identification, since it is preconfigured which pin is the wakeup on the embedded code, but for all other tests the second value represents the pin identification that corresponds to the multiplexer address in question.

To configure a test on the STM Nucleo pin PA0, which is the same location as A0 on the Arduino Uno. The pin mapping should be used to find the key values for the configuration. As can be seen below, next to PA0 is a number then a code that reads E#M#. The letter E stands for enable and M stands for multiplexer address. A configuration line to run Test #1 on PA0 would be the following in the configuration test file: 1,34,1,5, seen in Figure 9.0.3.

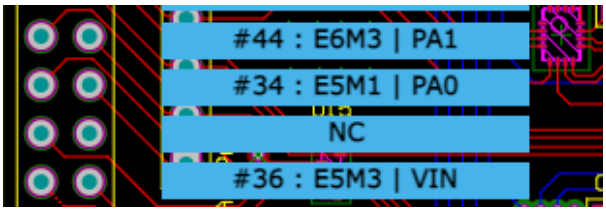


Figure 9.0.3: PA0 Pin Mapping Example

For the STM32 Nucleo low power mode tests, the STM has to be restarted to reconfigure the original state of the pins before it can be put into low power mode. This is likely due to the state of a pin being set to an output in a previous state but time constraints did not allow for the pursuit of a more rigid answer for that issue. If reconfigured it will likely need to be restarted in the same way.

Refer to **Appendix G** to connect a GPIO pin name to the Pin ID, Multiplexer Address and Multiplexer Enable.

Test #1 |
This test checks that the voltages produced by the subject development board reach a high and low logic level as specified in the configured range of the threshold configurations.

Test #2 |
This test confirms that voltages under a resistive load can reach a high and low logic level as specified in the configured range of the threshold configurations.

Test #3 |
This test confirms the operation of a pullup input and checks if the pull up resistance is within the configured range.

Test #4 |
When applicable, this test confirms the operation of a pulldown input and checks if the pull down resistance is within the configured range.

Test #5 |
This test confirms that the subject board's GPIO pins input can read high and low voltages.

Test #6 |
This test confirms that the subject board's ADC pins input can read voltages as specified in the configuration file.

Test #7 |
This test puts the board to sleep and checks that the difference in current is greater than the configured threshold in mA. In the current iteration, the wakeup pins are constrained to D2 or D3 on the Arduino board and PA0 on the STM Nucleo.

Test #8 |
Wakes up from power mode using the wakeup pin. In the current iteration, the wakeup pins are constrained to D2 or D3 on the Arduino board and PA0 on the STM Nucleo. The test also checks that the difference in current is greater than the configured threshold in mA.

Test #9 |
Resets the Subject Board Reset Pin

9.1 View Functions

The following functions are used to define the View module. This class serves to display data to the user from the Model class in the form of a graphic user interface (GUI). The view module has an update function which sends debugging data that is to be displayed, and a flag indicating whether a new test is ready to begin. There is also functionality to reset or clear the display information.

This module isolates all the graphic user interface (GUI) screen updates. If an alternative style is desired, changes to the functions of the View module must be made.

Below is a basic outline of the View class (**Figure 9.1**) along with function definitions.

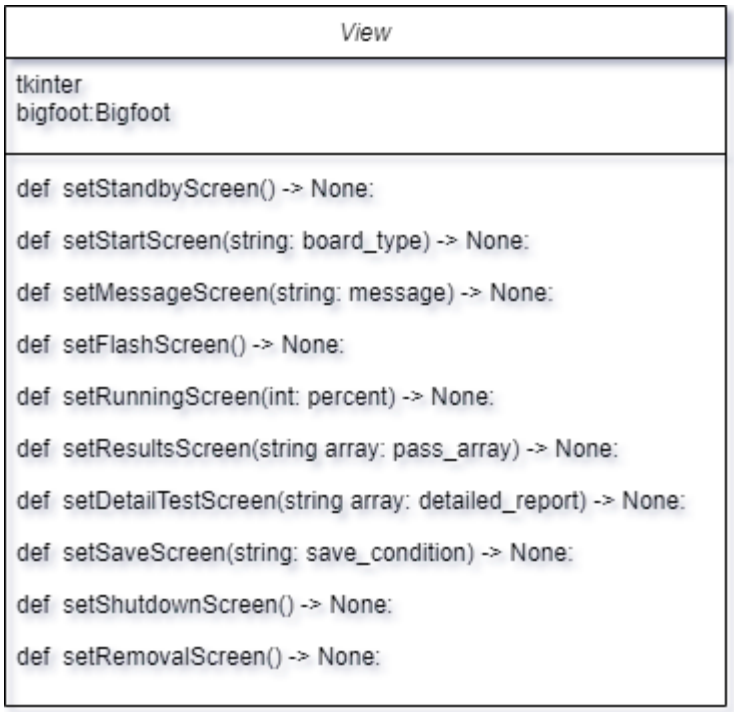


Figure 9.1: View Module Fields and Functions

The function prototypes are further described in **Appendix I**.

9.2 Model Functions

The Model module serves to manage the data taken from the Bigfoot module to the Controller module. It receives commands from the Controller module, and it receives its data from the subject board Bigfoot module which in turn handles the input and output signals to the Bigfoot PCB. The Raspberrian OS device communicates with the subject board to enable data transfer during testing. To accomplish this task, a flash test program is uploaded to the subject board. Once the upload is complete, UART commands are sent to and received from the subject board.

This module isolates the terminal call to the Raspbian OS (linux based OS). If an alternative embedded operating system is used, changes must be made to the Model module.

Below is a basic outline of the Model module (**Figure 9.2**) along with function definitions.

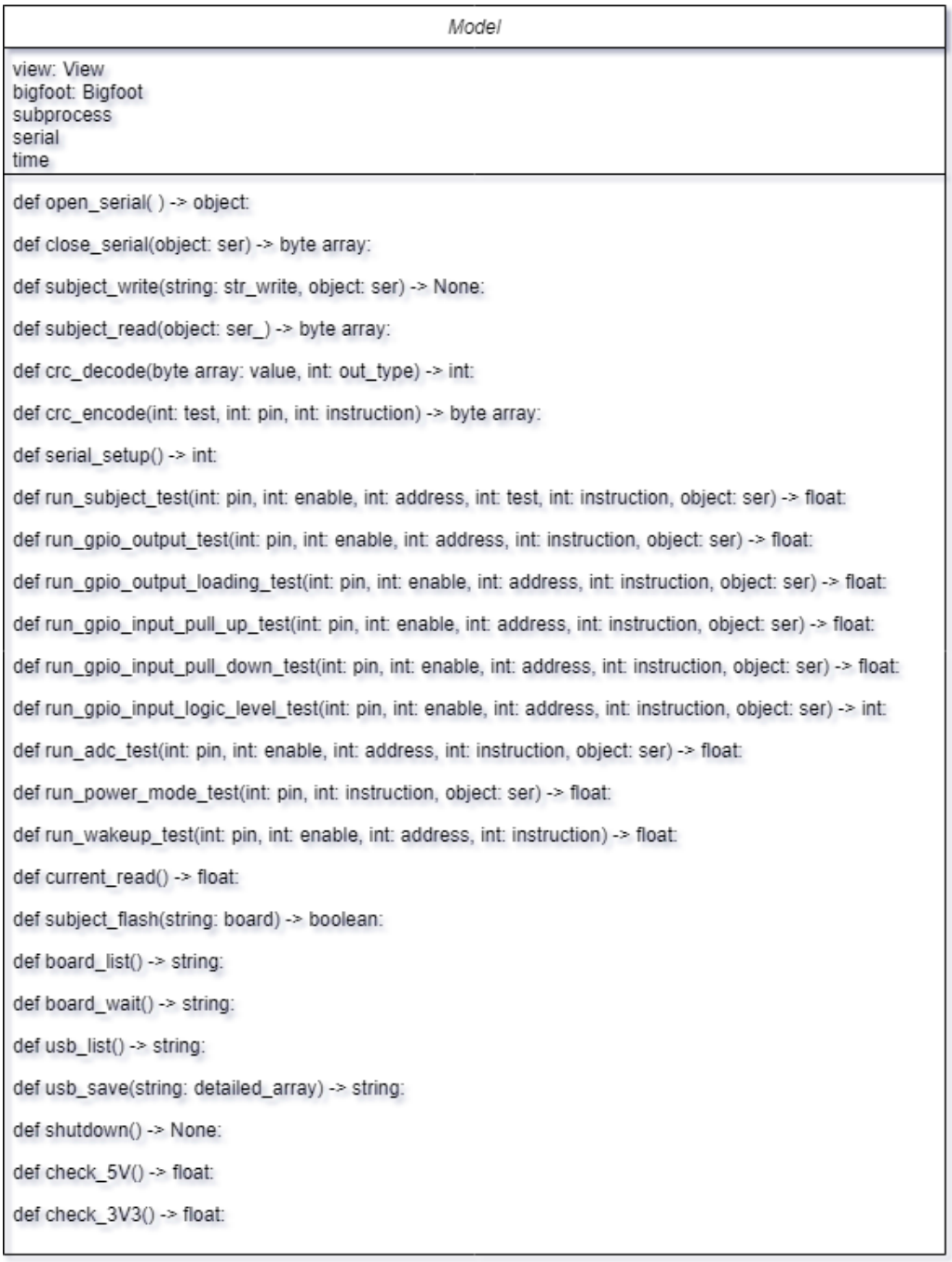


Figure 9.2: Model Module Fields and Functions

The function prototypes are further described in **Appendix J**.

9.3 Controller Functions

The following functions are used to define the Controller module. This module serves to handle state changes to the device, via the Bigfoot and Model modules. After interpreting the data from those modules, it is able to send updates to the view module along with the subject board driver programs.

This module isolates all the state changes and configuration file calls for the supported MicroDev tests. Any

alterations to the states or flow of the testing should be made from this module.

Below is a basic outline of the Controller class (**Figure 9.3**) along with function definitions.

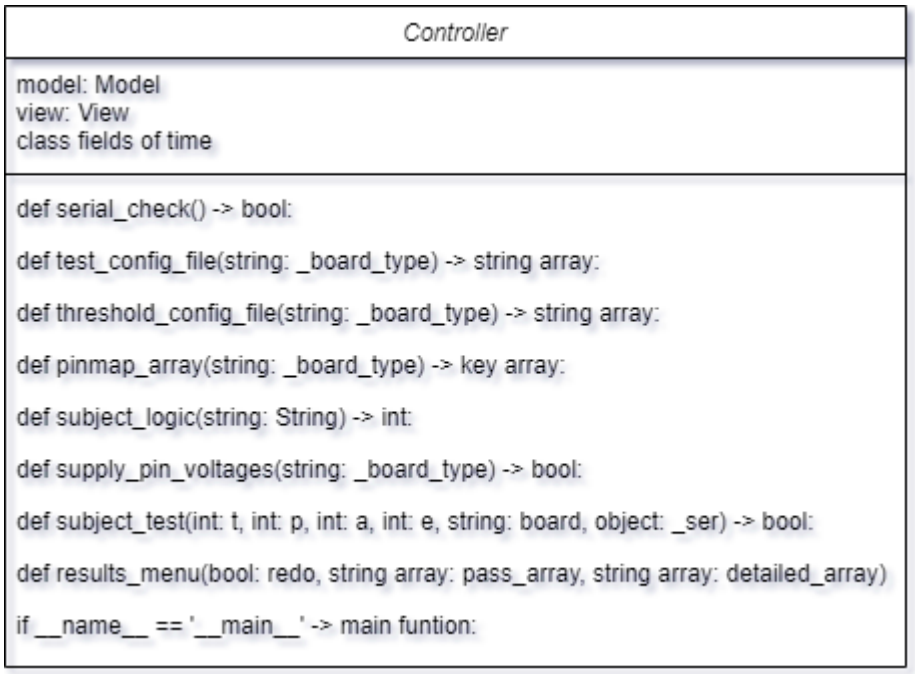


Figure 9.3: Controller Module Fields and Functions

The function prototypes are further described in **Appendix K**.

9.4 Bigfoot Functions

The Raspberry Pi communicates through I2C and digital signals with the **Bigfoot** PCB to enable the subject board GPIO to be analyzed and measured during testing. This module serves to handle inputs made by the user, via push buttons located on the enclosure. After interpreting these inputs, it is able to send updates to the Model module along with the subject board serial communication. The I2C communication includes an analog to digital converter (ADC), digital to analog convertor (DAC), and an INA current sensor.

This module was designed to isolate all digital signals. If an alternative device is used to replace the Raspberry Pi Zero this module would contain the Python code that needs to be altered to align with the GPIO and I2C of the alternative device.

The desired input and output signals are accomplished using the following functions:

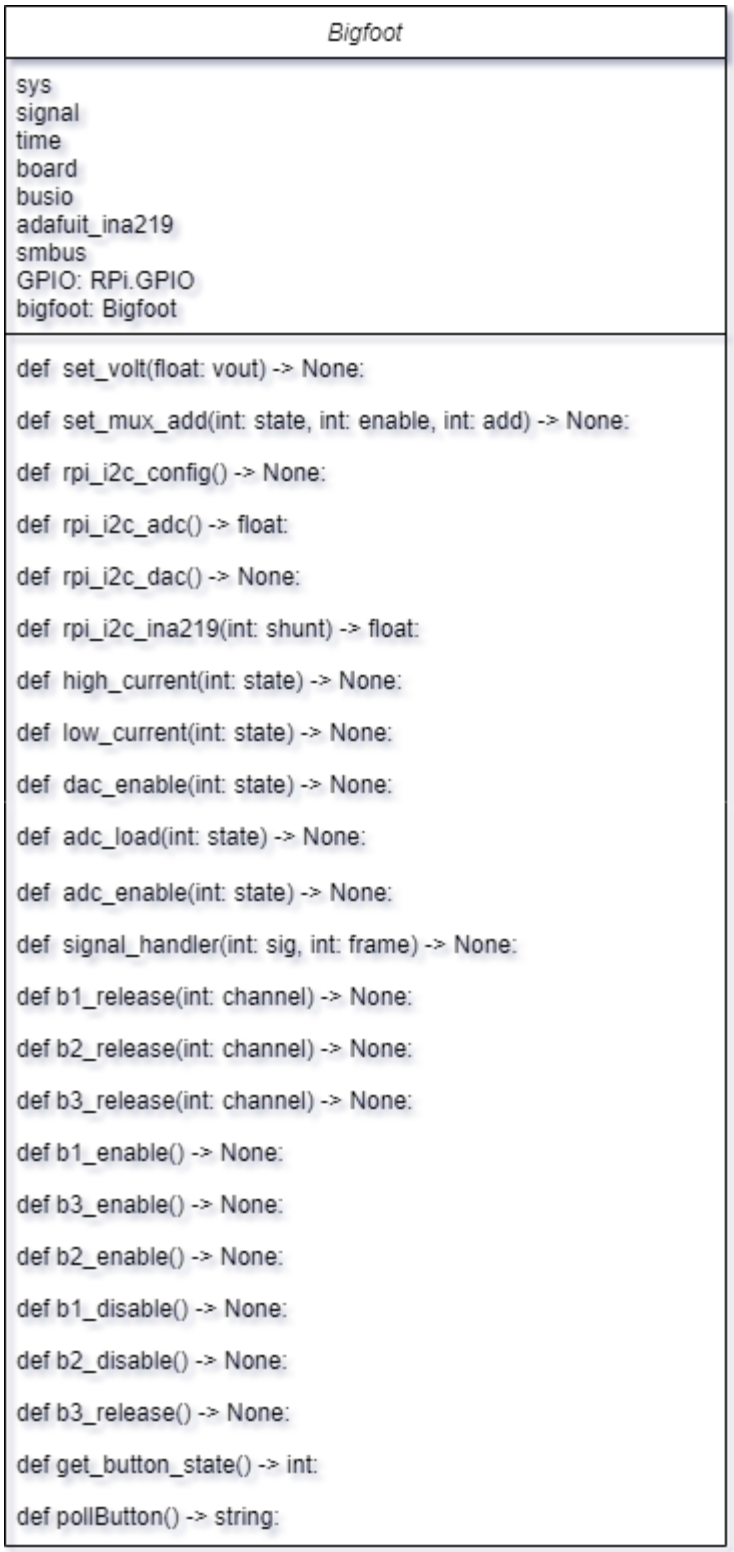


Figure 9.4: Bigfoot Module Fields and Functions

The function prototypes are further described in **Appendix H**.

10.0 Onboard Subject Board Testing Program

The program flashed to the subject board interacts directly with the program on the Raspberry Pi to carry out testing. The flowchart seen in **Figure 10.0** depicts how individual tests are facilitated and is common between the STM32 and Arduino programs along with the function declarations that can be seen in **Section 10.1**. This allows the testing to remain consistent across both subject boards and requires no modification to the software between the devices.

Validation testing for this section incorporates the facade testing architecture. This allows for the functions developed in both versions of this software to be tested in a simulated environment. These results are used to validate the functionality of the STM32's and Arduino's respective versions of the subject board testing program.

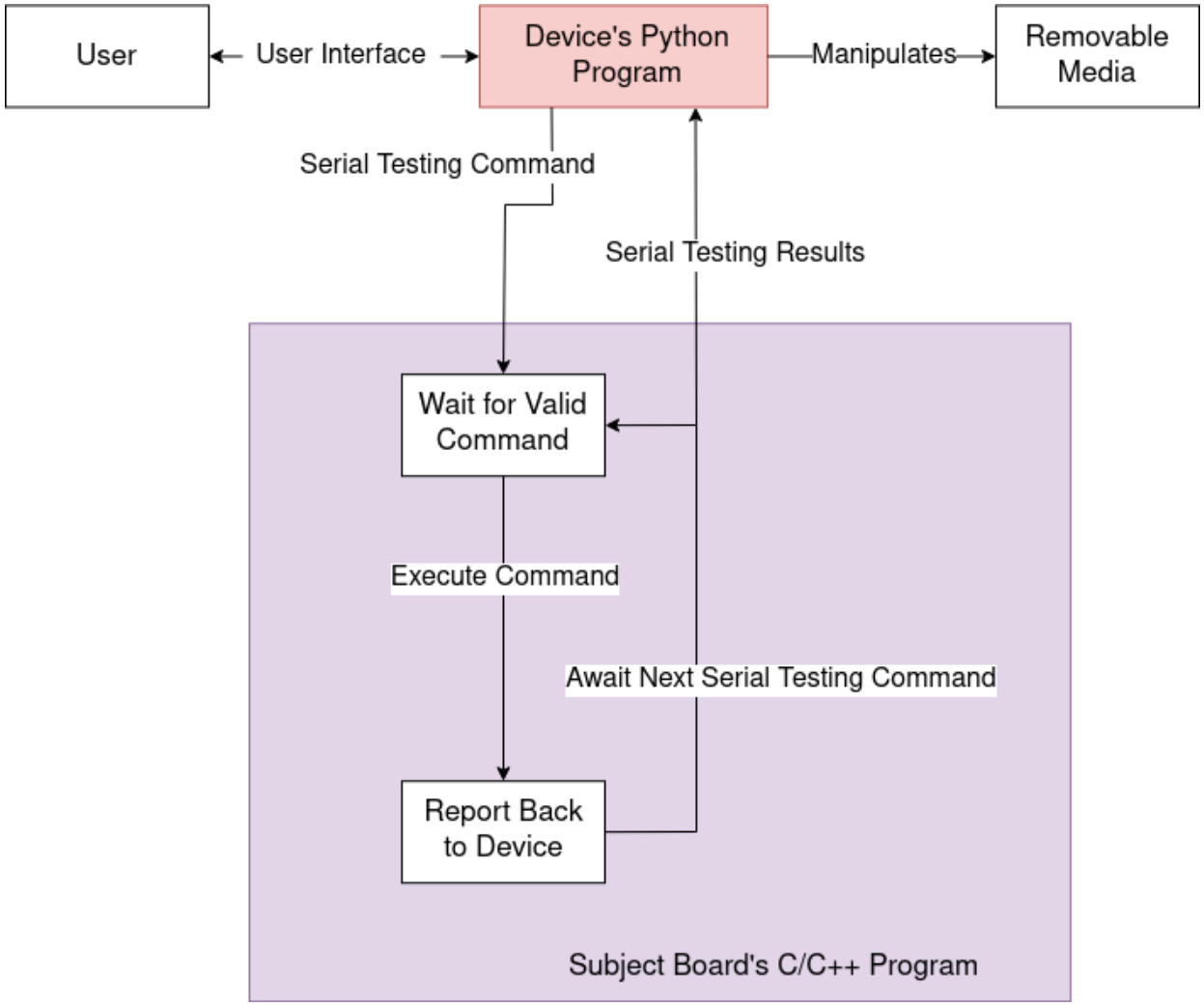


Figure 10.0: Subject Board Testing Flow Chart

A simple illustration of the software flow occurring between the MicroDev controller and the subject board can be seen in **Figure 10.0**. Flashing a testing program and sending commands to the subject board are important to accurately testing the GPIO and analog pins in isolation to each other. Confirmation to and from the subject board must be communicated to the central processor unit, the Raspberry Pi. Additionally, the protocol for the serial communication is shown in **Figure 10.1**.

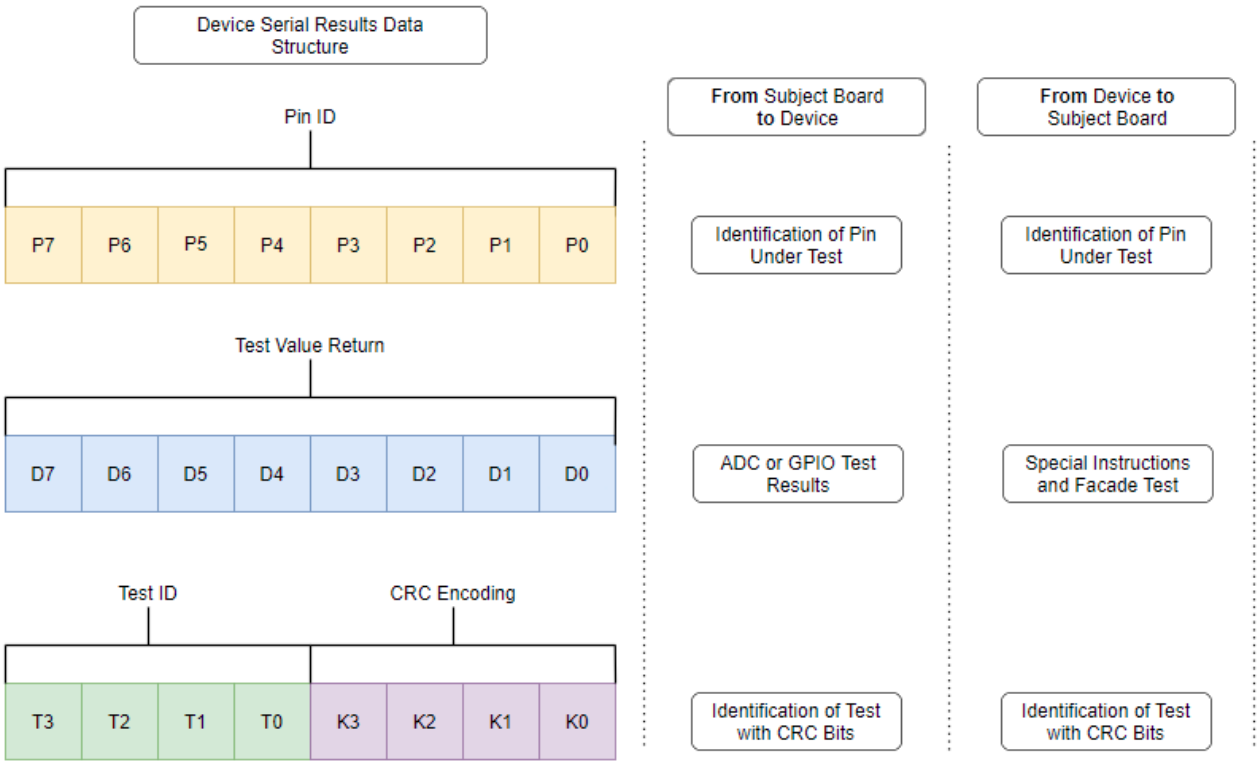


Figure 10.1: Serial Message Protocol

There are only a few commands that are required in order to decipher which test is occurring, which pin to isolate on the subject board, and whether a facade test is occurring. Due to this a fixed size was chosen to deliver and receive instructions to and from the subject board using serial communication. Identification of the specific test, specific pin, and test results can all be given through a 3 byte data packet. Error checking in the transmission of these packets will occur through using a Cyclic Redundancy Check (CRC) with four bits. Time delays and terminal calls are handled using the **time** and **subprocess** Python libraries. The Python Libraries are configured within the setup.sh file that is described in **Appendix F**. This serial communication process is illustrated in the flowchart below, **Figure 10.2**.

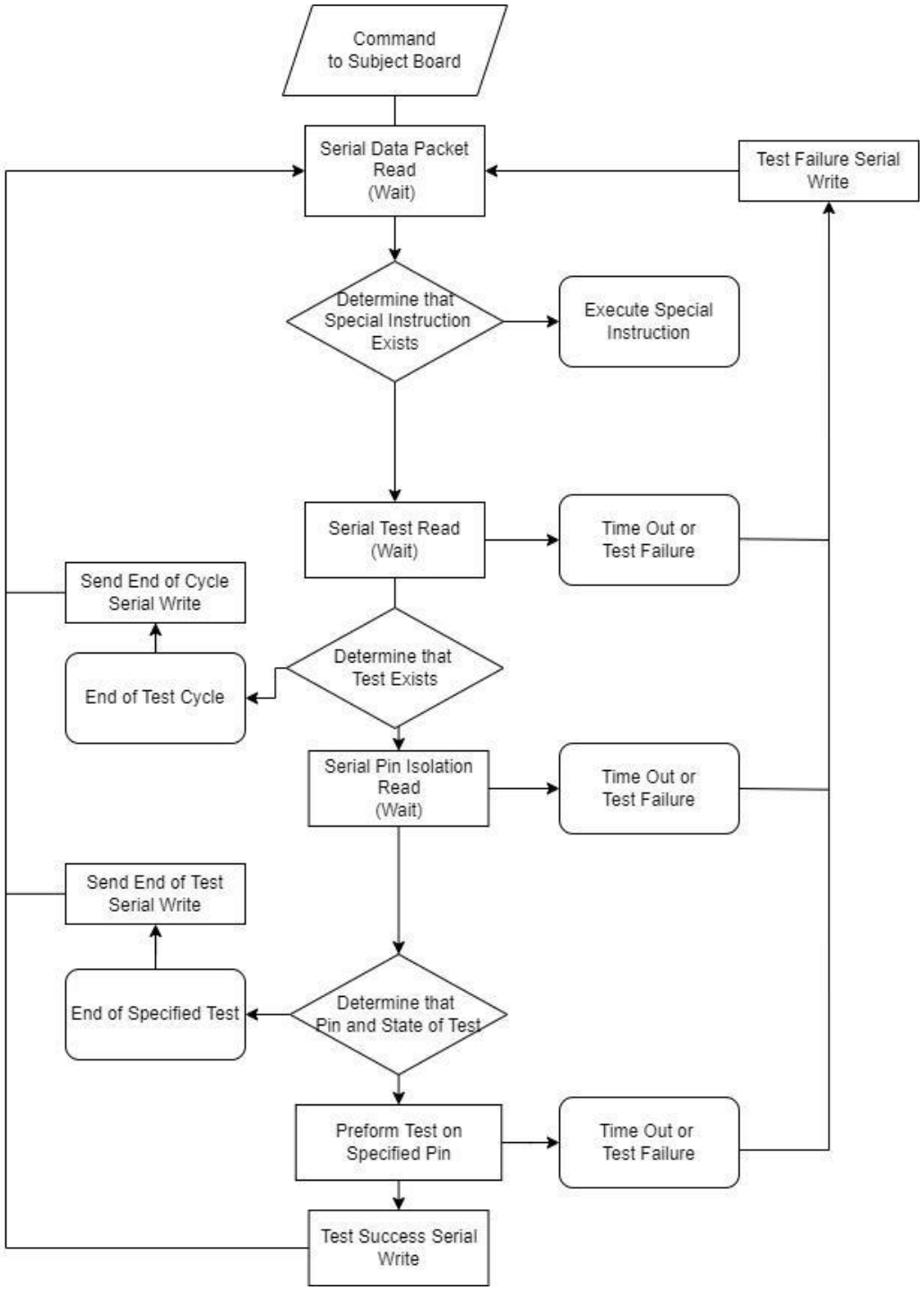


Figure 10.2: Subject Board Communication Flow Chart

10.1 Subject Board Testing Program Function Declarations

The subject board will have a test program for conducting necessary functions during a test cycle. This program will be in charge of setting up and initializing each pin register to its correct position during the course of the test. In order for each test to coincide with the setup on the subject board, there will also be serial communication between the subject board and the Raspberry Pi. The main loop will conduct serial communication in the subject board. Below are the function prototypes that set up the GPIO pins for each test. The function prototypes are further described in **Appendix L**.

In the case a new development board is chosen as the desired subject board please refer to **Appendix O**.

11.0 Graphical User Interface

The device utilizes a 3.5” LCD screen to display information to the user. This information includes instructions for loading and testing, and also displays labels for screen navigation. Additionally, the screen displays the test results for the operator to view once the test has concluded. Interaction with the device occurs via three momentary switches that allow the user to navigate through screens, save data, initialize a new test, cancel a running test, and initialize shut down procedures. The function of these switches changes as the state of the device changes. For example, in the “testing” state only 1 switch is functional, and when depressed, the testing process will cancel. Navigation of the various screens are described in detail below (shown in **Figure 11.1** & **Appendix D**). The illustrations below are for illustration purposes only and are not identical to the GUI screen.

There are some errors that may occur during edge cases with the Graphic User Interface state changes. For example, if the I2C communication or serial communication to the Bigfoot PCB is not connected, it can crash the program. In this case, the program will return to the Raspbian Desktop Screen and should be turned off and back on to bring it back to normal operation.

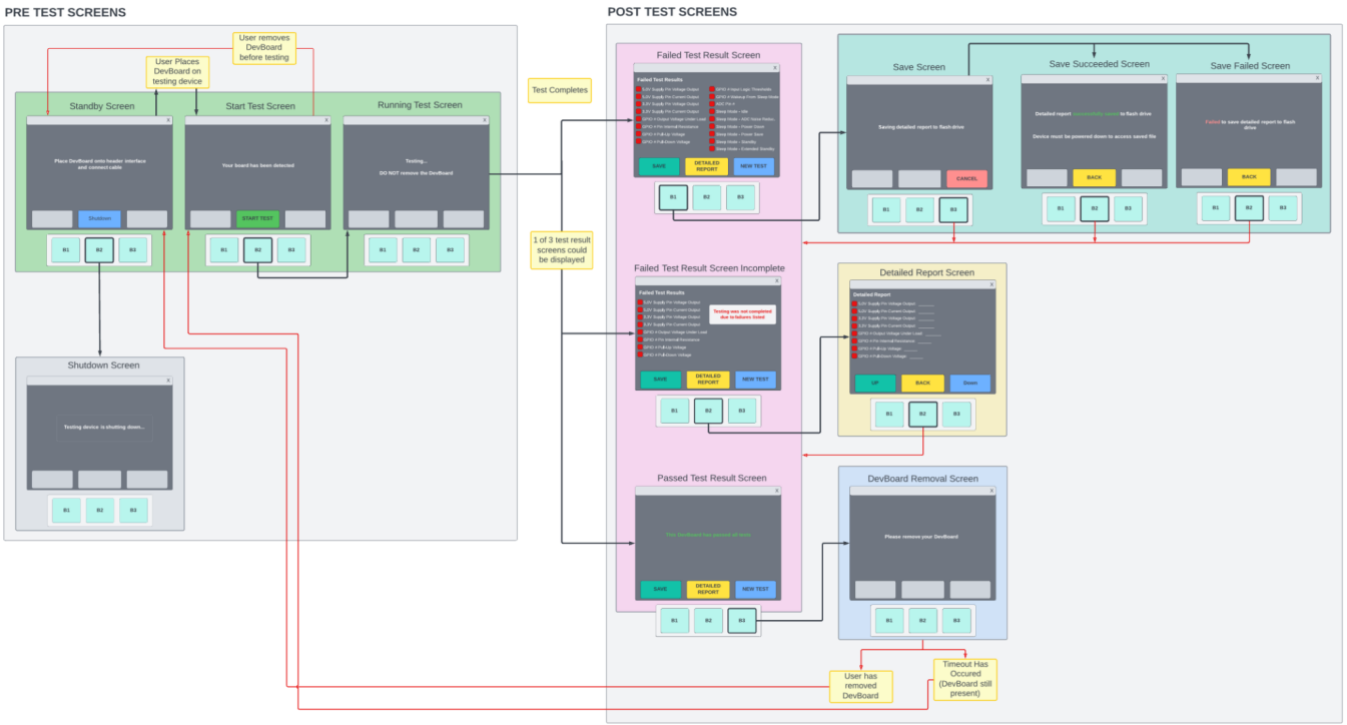


Figure 11.1: Navigation map of GUI screens

11.1 Standby / Shutdown Screens

When the device is powered on, not performing a test, and is not in sleep mode, it will display its standby screen. This screen displays a message to load a subject board, or press the center button to shut down the device. This screen can be seen below.

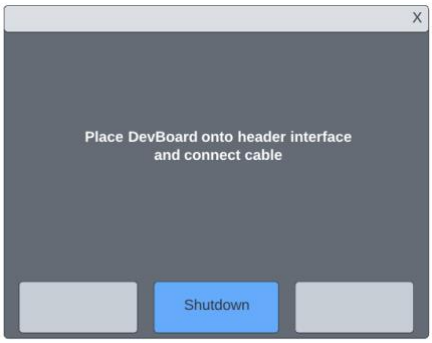


Figure 11.2: Standby Screen

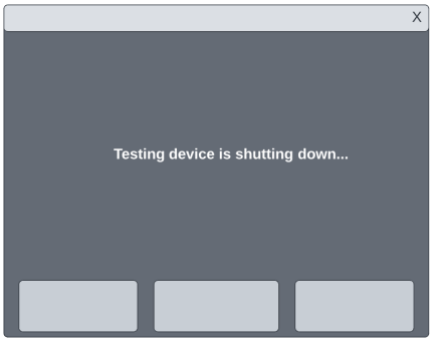


Figure 11.3: Shutdown Screen

11.2 Start Test Screen

The testing device will auto-detect when a subject board is loaded onto the header pin interface. The prompt notifies the user when this occurs and a "Start Test" button appears.

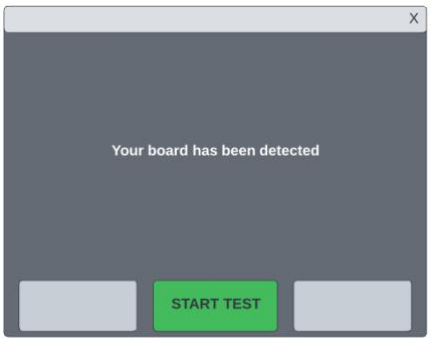


Figure 11.4: Start Test Screen

11.3 Testing Screen

Once the user presses the "Start Test" button, the testing will begin. This screen notifies the user that the test has begun and removal of the subject board is not advised.

Figure 11.5: Testing Screen

11.4 Passed Test Result Screen

Once the testing has concluded, one of two result screens will appear.

In the event that the subject board has passed all of the tests, a message will appear notifying the user.

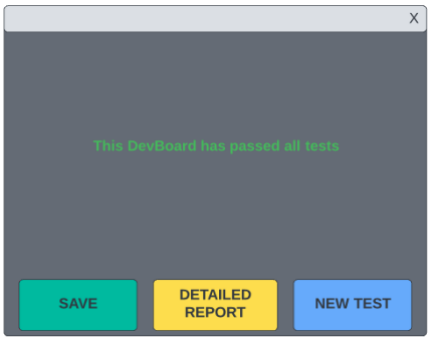


Figure 11.6: Passed Test Result Screen

11.5 Failed Test Result Screen

In the event one or more tests failed, the screen will display a list highlighting the functions which have not passed inspection. This list may be accompanied with a dialog box indicating that the test was ended prematurely by the testing device. In some cases this precaution is taken to prevent further damage to the subject board and/or the testing device. More details are shown in **Appendix H**.

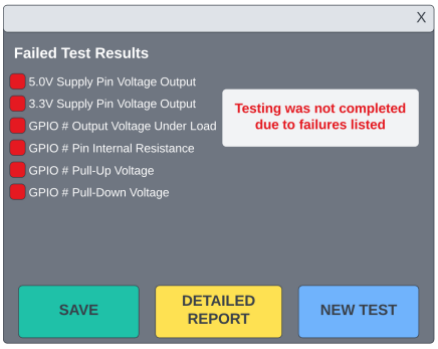


Figure 11.7: Failed Test Result Screen

11.6 Save Screens

The testing device offers a feature which allows the user to save the detailed test results of the subject board to the onboard Micro SD card. This option only comes available once a test has concluded, and its corresponding button is pressed. Once pressed, a series of screens may appear indicating different messages. The cancel button may be pressed while the device is attempting to save the data, and a back button can be pressed once saving has concluded to navigate back to the test result screen.

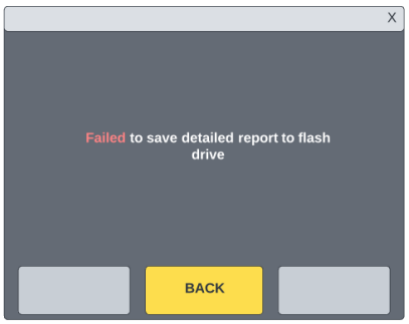


Figure 11.9: Save Failed Screen

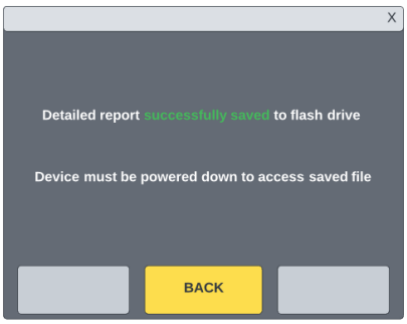


Figure 11.10: Save Succeeded Screen

11.7 Detailed Report Screen

The detailed report is available once testing has concluded and the “Detailed Report” button has been pressed. This screen reports the empirically collected data of the test including specific current and voltage measurements (Appendix H). If the list of failed test results is long, the “Up” and “Down” buttons can be pressed to traverse the list. The “Back” button can also be selected at any time to navigate back to the “Test Result Screen”.

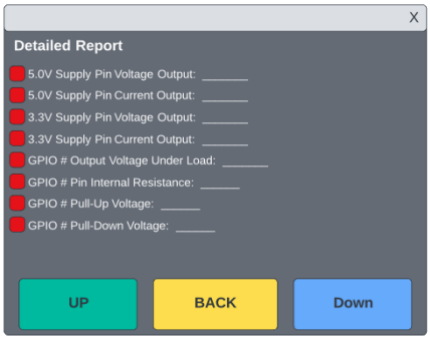


Figure 11.11: Detailed Report Screen

11.8 DevBoard Removal Screen

Additionally, from the “Test Result Screen” the user can select the button option for “New Test”. This button will prompt the user to remove the subject board from the device. If the subject board is not removed promptly, the device will move to the “Start Test Screen”. If the board is removed promptly, the device will return to the “Standby Screen” and wait for a new subject board to be interfaced.

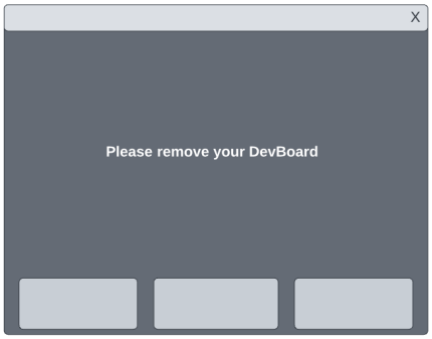


Figure 11.12: Removal Screen

11.9 Startup and Shutdown Procedures in Software

Proper startup and shutdown procedures are important to prevent the Linux based operating system on the Raspberry Pi from being corrupted. Startup of the Pi occurs automatically when power is connected and turned on. There is a mechanical switch that also can be switched on for the device to function (seen in **Figure 2.24**). The software design includes a press button option for shutdown after tests are completed.

12.0 Software Validation

The various functions above are validated through conducting unit tests on each given function. These are broken down into facade tests completed on the subject board, unit tests on linux python code, and finally validation of the integration between the subject board I/O and the control unit I/O.

For verification of the subject board pin mapping, facade code is utilized. These facade tests are purposed with confirming that the correct pins are tested for the subject board it is specified for and that it calls the correct function calls to complete the test. A simple flowchart of the commands and how they create a facade test state can be seen in **Figure 12.0**. Additionally, unit tests are available to confirm that hardware and software are correctly integrated. For each specification there is a corresponding test function and for each test there is a corresponding facade test.

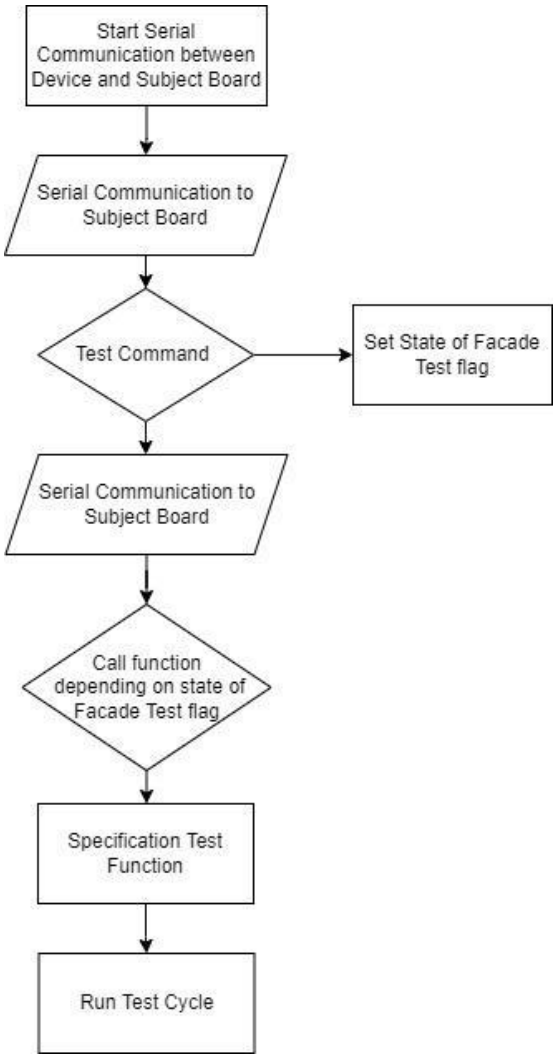


Figure 12.0: Subject Board Facade Test Flow Chart

These facade functions are designed to verify that the setup for the GPIO or analog pin is correctly mapped for each respective test. This is accomplished by returning the pin value corresponding with the identification value sent through serial communication. If the subject board returns the value that represents the register of the desired GPIO pin, that indicates that the pin is correctly mapped. The pin mapping details are illustrated in **Appendix G**.

PyCharm software on Linux Mint is recommended for conducting tests of the command line interface and serial communication to a desired subject development board. PyCharm allows Unit Tests to be run by simply clicking a green arrow that appears to the left of a unit test function. The unit tests relating to the *facade* pin mapping tests or serial communication tests are found in the Model_Test.py module. In order to isolate this module from the Bigfoot module, as to not cause errors, there is a blank version of the Bigfoot.py in the ~/*Python/Test/* file path that may be copied over into the main Python directory. The project file path ~*/Python/Integration/* contains the true Bigfoot module after testing is completed.

Conducting Tests on Pin Mapping of Development Board (Model_Test.py)

def test_run_gpio_output(self)
A general test that confirms the serial communication is established and the output test methods are not causing errors.

def test_gpio_uno_facade(self)
This test confirms that the register values for a given pin on the subject development board match the pin ID values on the Arduino Uno. This method works in conjunction with the configured tests in the uno_facadeTest.config of the local directory. In that file the first line is a description of the file and the following lines are facade tests. The file is comma delimited. The first value represents the pin ID and the second value is the expected register value associated with that GPIO pin.

def test_gpio_stm_facade(self)
This test confirms that the register values for a given pin on the subject development board match the pin ID values on the STM32 Nucleo. This method works in conjunction with the configured tests in the stm_facadeTest.config of the local directory. In that file the first line is a description of the file and the following lines are facade tests. The file is comma delimited. The first value represents the pin ID and the second value is the expected register value associated with that GPIO pin.

Conducting Tests on Serial Communication (Model_Test.py)

def test_serial_basic(self)
Runs a unit test that confirms current operation of the serial communication to and from any subject board. A passing test means that basic serial communication is operational.

def test_crc_encode(self)
Runs a unit test that confirms correct encoding of a 3 byte data packet.

def test_crc_decode(self)
Runs a unit test that confirms correct decoding of a 3 byte data packet.

Conducting Tests Via Inspection for CLI (Model_Test.py)

def test_board_list_None(self)
Tests that the state in which nothing is connected is detectable.

def test_board_list_Uno(self)
This test requires an Arduino Uno to be connected through a USB port. It tests the state in which an Arduino Uno is detectable.

def test_board_list_STM32F411(self)
This test requires an STM32F411RE to be connected through a USB port. It tests the state in which an STM32F411RE is detectable.

def test_board_list_STM32F401(self)
This test requires an STM32F401RE to be connected through a USB port. It tests the state in which an

STM32F401RE is detectable.

def test_board_list_STM32F446(self)
This test requires an STM32F446RE to be connected through a USB port. It tests the state in which an STM32F446RE is detectable.

def test_board_flash(self)
This unit test confirms that the board can be auto detected and flash a program successfully. The results on the console must be inspected to confirm the operation was successful.

def test_usb(self)
Runs a test that indicates if the USB is detected currently within the Python code.

The **Geany** application on the Raspbian OS is recommended for running validation tests for software integration. Geany is a lightweight programming environment that can run relatively quickly on even the original Raspberry Pi Zero W. Tests can be found in the Controller_Test.py module that correlate with each of the specifications found in **Table 13** below. Within the Controller_Test module, simply uncomment the test(s) to run your desired validation test and run the module as the main.

The validation tests are meant to be done with all the hardware connected and the Arduino Uno inserted for a test.

def Validation_3021(pin, enable, address
Configures an output test that sets a voltage that remains on until enter is pressed to exit the test.

def Validation_3031(pin, enable, address)
Configures a pull-up resistor test that sets a voltage that remains on until enter is pressed to exit the test.

def Validation_3032(pin, enable, address)
Configures a pull down resistor test that sets a voltage that remains on until enter is pressed to exit the test.

def Validation_3061_3V3_Logic(pin, enable, address)
Configures the Bigfoot PCB to deliver a voltage to the subject board. The subject board then reads and writes back the measured value of its high or low logic measurement. Remains paused until enter is pressed to exit the test.

def Validation_3071_3V3_Logic(pin, enable, address)
Configures the Bigfoot PCB to deliver a voltage to the subject board. The subject board then reads and writes back the measured value of its ADC measurement. Remains paused until enter is pressed to exit the test.

def Validation_3081(pin, instruction)
Configures a power mode that remains on until enter is pressed to exit the test.

def Validation_3091(pin, enable, address)
Configures and wakes up the subject board by sending a signal to the desired pin.

def Littlefoot_Test()
Configures every multiplexer path with a 3.3 volt signal, each with a pause in between. This test streamlines validation of the Littlefoot PCB hardware.

Table 13. Software validation methods

Specification	Method of Validation
3.00	This will be confirmed by performing a test cycle that corresponds with the thresholds on the 5V and 3.3V voltage supplies.
3.02	The subject board and MicroDev hardware will be connected for the Validation_3021 function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that corresponds with the thresholds on each GPIO pin while under simulated load sourcing.
3.03.1	The subject board and MicroDev hardware will be connected for the Validation_3031 function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that corresponds with a simulated pull-up input setting on the subject board.
3.03.2	The subject board and MicroDev hardware will be connected for the Validation_3032 function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that corresponds with a simulated pull-down input setting on the subject board. This test will not take place on the Arduino Uno since it does not have this type of functionality.
3.06	The subject board and MicroDev hardware will be connected for the Validation_3061_3V3_Logic function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that corresponds with setting each GPIO pin to inputs and waiting for a signal.
3.07	The subject board and MicroDev hardware will be connected for the Validation_3071_3V3_Logic function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that corresponds with setting each ADC pin to inputs and waiting for a signal.
3.08	The subject board and MicroDev hardware will be connected for the Validation_3081 function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that adequately sets each sleep mode and preconfigured wakeup pin in the correct order.
3.09	The subject board and MicroDev hardware will be connected for the Validation_3091 function. This function will confirm the various output calls made from the subject board are adequate to perform a test cycle that adequately sets each sleep mode and preconfigured wakeup pin in the correct order.

The Graphical User Interface (GUI) functionality is verified via inspection for every use case.

Inspection of the GUI and Tests occurred by connecting the 3V3 output and 5V output pin to the MicroDev header fixture, and then running the test while only one pin was connected to the header fixture. This method of inspection allowed us to confirm that each test failed besides the tests utilizing the pin in question. Then, by running through each screen during the test, it was confirmed that the altered state of the testing does not incidentally create an edge case that would cause unexpected issues.

Table 14. GUI validation methods

Specification	Method of Validation
4.00.1	The subject board's identification information is verified through inspection that the identification information is shown and is correctly represented on the graphical user interface.

4.01.1	The labels will be identified through inspecting the buttons. The accuracy of the labels is determined through using the user interface. Start is to be pressed to determine if the start button works, scrolling up and scrolling down is also to be used to determine their functionality.
4.02.1	The “begin test” button is tested through visual inspection of its ability to run a test and display each possible result using test software.
4.03.1	Through inspection of the results of predetermined subject board tests, the device's ability to report at the conclusion of testing will be verified.
4.03.2	The device's ability to report each error will be verified through inspection of the results of predetermined subject board tests.
4.04.1	The removable media will be verified through inspection that the configuration values are shown and are correctly represented on the graphical user interface (format shown in **Appendix H)**.
4.05.1	Commented software consists of a description of each function and a listing of its inputs and outputs. This validation is accomplished through inspection of the code for this feature.
4.05.2	This specification is satisfied by the development of the software portion of the final manual. This manual consists of a description of each function as well as required updates in the case of new subject boards that may be added in future iterations. These updates may include changes to a configuration file, subject board program, or setup of the Raspberry Pi OS with device software.

Discussion of Engineering Responsibilities

In order to mitigate potential negative consequences that may result from our team's decisions, this section discusses the global, cultural, social, environmental, and economic factors that are within the scope of this project. This project remains within well established engineering practices and is not at risk of creating a new market for unethical practices in manufacturing. The custom PCB design will be manufactured in China. The practices surrounding the mining and refining of Rare Earth Minerals in China has been a topic of environmental and humanitarian concern that should not just be ignored. There are also other environmental concerns generally associated with China but a PCB design that uses standard materials is not known to have a significant economic or environmental impact. Although arguable, it is thought that since 1979, trade between the U.S. and China is partially responsible for resuscitating the Chinese economy and increasing the standard of life for an average individual in China. Under that conclusion, along with our present knowledge of Chinese manufacturing, using China as a source for manufacturing does not contribute to poor working conditions that may exist in China.

This project opens up the plans for replicating a testing fixture to a broader audience, such as other universities. It will also make available a PCB design and 3D printable enclosure that will make it relatively simple for people to take and customize to their own purposes. This could help benchmark concepts for people to develop a better academic understanding. Alternatively, having the means to debug a microcontroller device with even less critical analysis might create a social or cultural environment that does not facilitate developing certain analytical skills that students would have otherwise learned. On the other hand, some students have more background knowledge due to resources that were made available due to wealthy contributors in their community and other students come in having to make up that gap in knowledge on their own. The MicroDev device can assist in freeing up bandwidth for incoming students as they learn the fundamentals of embedded systems. The former issue is best addressed by restricting access to the device to 3rd and 4th year engineering students because at that point in the engineering program they should have learned to debug their subject board on their own.

Conclusion

In conclusion, the MicroDev testing fixture is designed to test the functionality of the specific subject board's vital hardware. These tests include testing the subject board's GPIO pins, ADC, logic levels, voltage supply pins, and current draw. The device was designed to be easily reproduced by other universities. The enclosure design can be reproduced using the published CAD files and a 3D printer. Code and configuration files will be shared publicly which can be uploaded into a Raspberry Pi. PCB layout can be replicated using the published Altium schematic and PCB layout files. This project opens the door of developing testing fixtures to a broader audience and therefore creates the potential for a broader use of this type of device.

There are some constraints in the ability to reconfigure the project. For the STM sleep mode tests, the STM has to be restarted to reconfigure the original state of the pins before it can be put into low power mode. This is likely due to the state of a pin being set to an output in a previous state but time constraints did not allow for the pursuit of a more rigid answer for that issue. There is also some room for improvement in how the digital input test is handled, since it allows for the potential for false positives. In a redesign of the software, one might handle the problem of testing floating inputs by not running this test upon failures of the previous input pullup and input pulldown tests. In a redesign of the software, the MUXs should be replaced if the goal is to populate boards by hand since they proved to be difficult in hand soldering. The TVS diodes may also cause issues with testing if left on.

Appendix A - Bill of Materials

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	TOTAL QTY.	
CNC BOM					
1	MDTM0001	Base Plate	1	10	IN HOUSE
2	MDTM1002	Large Enclosure Top Plate	1	10	
4	MDTM2001	Small Enclosure Top Cover	1	10	
LASER CUTTING BOM					
3	MDTM1005	ESD GROUNDING PLATE	1	10	IN HOUSE
2	MDTM0003	Name Plate	1	10	

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	TOTAL QTY.	Print Time	Total Print Time (min)	Material (grams)	Printer	Material
3D PRINT BOM - (download files at https://github.com/hancheyn/MicroDev/tree/main/MDT0000_STLs_V20:)									
1	MDTM0000	Main enclosure	1	10	37:30:00	22500	385	Ultimaker S5	PLA
2	MDTM0002	Crank lever hard stop block	2	20	0:06:00	240	1	Ender 3 S1 Pro	PLA
3	MDTM1004	Rear storage pocket	1	10	4:30:00	2700	31	Ender 3 V2	PLA
4	MDTM1006	USB hub bracket	1	10	1:09:00	690	7	Ender 3 V2	PLA
5	MDTM1008	Button Panel Cover	1	10	0:33:00	330	3	MarkForged Onyx Pro	Nylon
6	MDTM1101	LCD Housing Hinge Base	1	10	1:19:00	790	8	MarkForged Onyx Pro	Nylon
7	MDTM1102	LCD Housing	1	10	3:54:00	2340	26	MarkForged Onyx Pro	Nylon
8	MDTM1103	LCD Retainer Bezel	1	10	1:14:00	740	8	MarkForged Onyx Pro	Nylon
9	MDTM1104	LCD Power Button Extension	1	10	0:02:00	20	0.25	Ender 3 S1 Pro	PLA
10	MDTM1201	USB Slide Housing	1	10	0:55:00	550	6	MarkForged Onyx Pro	Nylon
11	MDTM1202	USB Slider	1	10	0:29:00	290	3	MarkForged Onyx Pro	Nylon
12	MDTM2003	Left header cover - Ard	1	10	0:43:00	430	4	MarkForged Onyx Pro	Nylon
13	MDTM2003	Left Header Cover - STM	1	10	1:01:00	610	6	MarkForged Onyx Pro	Nylon
14	MDTM2004	Right Header Cover - Ard	1	10	0:43:00	430	4	MarkForged Onyx Pro	Nylon
15	MDTM2004	Right Header Cover - STM	1	10	1:01:00	610	6	MarkForged Onyx Pro	Nylon
16	MDTM2005	Rear alignment guide	1	10	0:35:00	350	4	Ender 3 S1 Pro	PLA
17	MDTM3001	PCB Mounting Plate	1	10	2:11:00	1310	16	Ender 3 V2	PLA
18	MDTM3002	Ribbon cable retention bracket	1	10	0:31:00	310	3	Ender 3 S1 Pro	PLA
19	MDTM4001	Lower race	1	10	1:54:00	1140	13	MarkForged Onyx Pro	Carbon Fiber Nylon /cnt. Fiber
20	MDTM4002	Upper race	1	10	2:12:00	1320	13	MarkForged Onyx Pro	Carbon Fiber Nylon /cnt. Fiber
21	MDTM4003	Crank	1	10	2:07:00	1270	13	MarkForged Onyx Pro	Carbon Fiber Nylon /cnt. Fiber
22	MDTM4004	Merged link	1	10	0:53:00	530	5	MarkForged Onyx Pro	Nylon
23	MDTM4005	Push rod base	2	20	0:31:00	1240	3	MarkForged Onyx Pro	Nylon
24	MDTM4007	Push rod Hat	2	20	0:14:00	560	1	MarkForged Onyx Pro	Nylon
25	MDTM4008	Crank Handle	1	10	2:23:00	1430	20	MarkForged Onyx Pro	Nylon
26	MDTM4009	Crank Retaining Washer	1	10	0:05:00	50	0.5	Ender 3 S1 Pro	PLA
27	MDTM5001	SDCard_Protector	1	10	0:12:00	120	1	Ender 3 S1 Pro	PLA
								Print Time Total:	715.00 Hrs

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
RASPBERRY PI / LCD BOM											
1	Pi Zero	Pi Zero 2W MSRP	1	TBD	TBD		$15.00	1	$15.00	8	$120.00
2	Pi4-2016011800	(O) 3.5" LCD	1	Amazon	https://www.amaz		$32.18	1	$32.18	10	$321.80
3	SDSDQUA-032G	(P) SanDisk 32GB Ultra microSDHC Card	1	Amazon	https://www.amaz		$8.41	1	$8.41	8	$67.28
								Unit Total:	$55.59	Total:	$509.08

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
CABLES											
1	4298	(A) Pi Power Supply USB-C	1	Adafruit	https://www.adafru	1	$7.95	1	$7.95	10	$79.50
2	4052	(C) Snap Mount USB C	1	Adafruit	https://www.adafru	1	$5.95	1	$5.95	10	$59.50
3	3610	(D) USB Micro to USB Micro	0.5	Adafruit	https://www.adafru	1	$1.95	0.5	$0.98	5	$9.75
4	908	(E) Panel Mount USB A	2	Adafruit	https://www.adafru	1	$3.56	2	$7.12	20	$71.20
5	C1419	(F) 4 Port USB Hub	1	ChicagoELEDist	https://chicagodist	1	$5.99	1	$5.99	10	$59.90
6	4055	(G) Snap Mount USB A	2	Adafruit	https://www.adafru	1	$4.46	2	$8.92	20	$89.20
7	2819	(H) HDMI Adapter	1	Adafruit	https://www.adafru	1	$2.66	1	$2.66	10	$26.60
8	4054	(I) Snap Mount HDMI	1	Adafruit	https://www.adafru	1	$7.95	1	$7.95	9	$71.55
9	2197	(J) HDMI Flat Cable	1	Adafruit	https://www.adafru	1	$3.56	1	$3.56	10	$35.60
10	FE-USB-ACB590-6IN	(K) USB-A to 90* Micro	1	MyCableMart	https://www.mycal	1	$2.23	1	$2.23	10	$22.30
11	13764792	(L) Male USB mini to USB Micro	1	Amazon	https://www.amaz	1	$5.99	1	$5.99	10	$59.90
12		(M) USB-A to 90* USB-A	1	Amazon /Startech	https://www.amaz	1	$6.97	1	$6.97	10	$69.70
13	0192030483	(Q) Ring Terminal Connector for ESD GND Plate	1	Digikey	https://www.digike	1	$0.12	1	$0.12	10	$1.18
15	4170	(R) 2x8 Ribbon Cable	1	Adafruit	https://www.adafru	1	$1.76	1	$1.76	10	$17.60
16	7849K31	(S) Power Switch Terminals	3	McMaster	Insulated Heat-Shr	25	$15.42	0.12	$1.85	2	$30.84
								Unit Total:	$68.14	Total:	$673.48

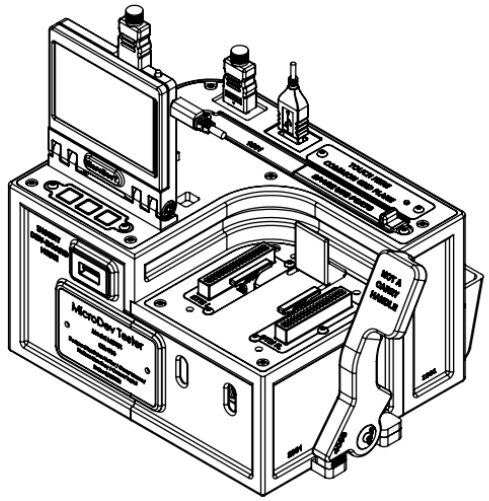
ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
SWITCHES											
1	1010	(N) 12mm Square Tactile Button Switch	3	Adafruit	https://www.adafru	15	$5.95	0.2	$1.19	2	$11.90
2	RB141C1000-114-ND	(B) Main Power Toggle Switch	1	Digi-Key	https://www.digike	1	$1.42	1	$1.42	12	$17.03
								Unit Total:	$2.61	Total:	$28.93

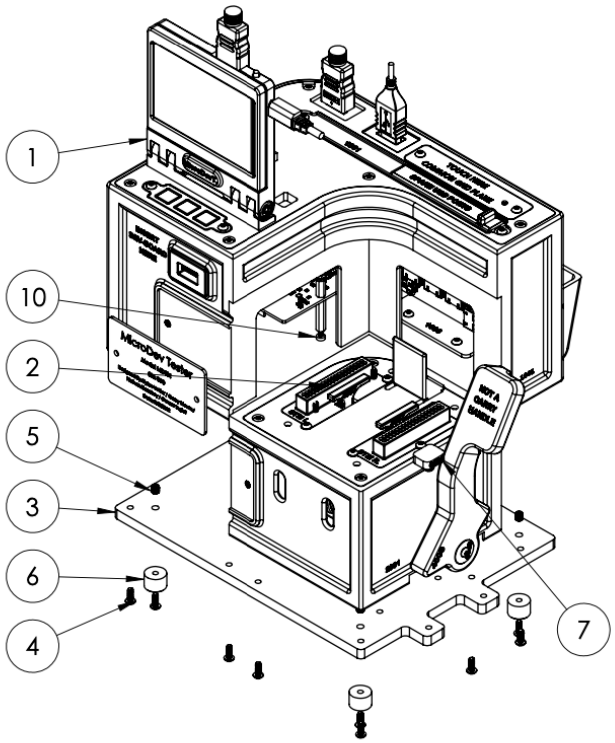
ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
PCB BOM											
1	C0805C104M3RAC7800	Generic 0805 MLCC Capacitor - 0.1 uF	13	Digikey	https://www.digike	1	$0.03	13	$0.36	156	$4.29
2	CL21B105KAFNNNE	Generic 0805 MLCC Capacitor - 1 uF	9	Digikey	https://www.digike	1	$0.03	9	$0.29	108	$3.42
3	CL21A106KAYNNNE	Generic 0805 MLCC Capacitor - 10 uF	6	Digikey	https://www.digike	1	$0.08	6	$0.48	72	$5.70
4	UVR0J101MDD6TP	Aluminum Electrolytic Capacitor 0.059" Lead Spacing, 0.157" Diameter - 100 uF	2	Digikey	https://www.digike	1	$0.25	2	$0.50	24	$6.05
5	ILBB0805ER301V	IND, Chip Ferrite Bead, 330 Ohm, +/- 25%, 1.5 A, -55 to 125 degC, 0805 (2012 Metric), Tape and Reel	3	Digikey	https://www.digike	1	$0.07	3	$0.22	36	$2.68
6	USB1130-15-A	USB A 2.0 CONNECTOR THRU HOLE RIGHT ANGLE	2	Digikey	https://www.digike	1	$0.66	2	$1.33	24	$15.93
7	FQP17P06	MOSFET P-CH 60V 17A	2	Mouser	https://www.mous	1	$1.73	2	$3.46	24	$41.52
8	RMCF0805ZT0R00	Generic R0805 Chip Resistor - 0 Ohm	15	Digikey	https://www.digike	1	$0.01	15	$0.11	180	$1.28
9	RNCP0805FTD2K00	Generic R0805 Chip Resistor - 2 kOhm	3	Mouser	https://www.mous	1	$0.09	3	$0.26	36	$3.06
10	RK73H2ATTD3400F	Generic R0805 Chip Resistor - 390 Ohm (minimum 5% tol)	1	Digikey	https://www.digike	1	$0.07	1	$0.07	12	$0.80
11	RK73B2ATTD204J	Generic R0805 Chip Resistor - 200 kOhm	1	Digikey	https://www.digike	1	$0.07	1	$0.07	12	$0.86
12	WSL0805R0400FEA18	Generic R0805 Chip Resistor - 0.04 Ohm	1	Mouser	https://www.mous	1	$0.81	1	$0.81	12	$9.77
13	RK73H2ATTD40R2F	Generic R0805 Chip Resistor - 40.2 Ohm	1	Mouser	https://www.mous	1	$0.04	1	$0.04	12	$0.48
14	RNCP0805FTD10R0	Generic R0805 Chip Resistor - 10 Ohm	2	Digikey	https://www.digike	1	$0.07	2	$0.15	24	$1.75
15	SG73S2ATTD472J	Generic R0805 Chip Resistor - 4.7 kOhm	4	Digikey	https://www.digike	1	$0.04	4	$0.17	48	$2.07
16	MCP3021A4T-E/OT	10 BIT ADC I2C SINGLE CHANNEL	1	Digikey	https://www.digike	1	$1.42	1	$1.42	12	$17.04
17	SN74LVC1G66DRLR	1:1 1.65-5.5V Bidirectional Analog Switch	3	Digikey	https://www.digike	1	$0.43	3	$1.29	36	$15.49
18	DAC121C081CISD/NOPB	12 BIT DAC I2C 2.7-5.5V SUPPLY WITH SUPPLY REFERENCE	1	Mouser	https://www.mous	1	$2.85	1	$2.85	12	$34.20
19	INA219BID	12 BIT I2C CURRENT/VOLTAGE MONITOR	1	Texas Instruments	https://www.ti.com	1	$2.26	1	$2.26	12	$27.07
20	LSF0102DCTR	Dual Bidirectional Multi-Voltage Level Translator 8-SM8 -40 to 125	1	Digikey	https://www.digike	1	$0.67	1	$0.67	12	$7.98
21	74AXP8T245PWJ	Translation - Voltage Levels ANALOG & LOGIC ICS (3V3 to 5V)	1	Digikey	https://www.digike	1	$0.83	1	$0.83	12	$10.00
22	TPD2E001DRYR	TVS DIODE 2 CHANNEL ESD FOR HIGH SPEED INTERFACE	1	Digikey	https://www.digike	1	$0.56	1	$0.56	12	$6.72
23	TMUX1208QRSVRQ1	8:1 5V MUX w/1.8V logic	8	Digikey	https://www.digike	1	$1.35	8	$10.80	96	$129.62
24	D3V3F8U9LP3810-7	8 CHANNEL TVS DIODE 5.5V BREAKDOWN 5V CLAMP	8	Digikey	https://www.digike	1	$0.22	8	$1.76	96	$21.06
25		Female-Male: **1x6** (Little foot Header **Extension** for **Arduino**)	1								
26	85	Female-Male: **1x8** (Little foot Header **Extension** for **Arduino**)	2	Adafruit	https://www.adafr	1	$1.76	1	$1.76	12	$21.12
27		Female-Male: **1x10** (Little foot Header **Extension** for **Arduino**)	1								
28	4079	Female - Male: 2x19 (Little foot Header Extension for STM)	2	Adafruit	https://www.adafr	1	$1.95	2	$3.90	24	$46.80
29	2222	Male - Female: 2x19 (Little foot Thru-Hole for STM)	2	Adafruit	https://www.adafr	1	$1.50	3	$4.50	36	$54.00
30		Male - Female: 2x20 (Big foot Thru-Hole for Rasp Pi)	1								
31		Male Header: **1x4** (Thru-Hole for 3 BTNs / 1 GND)	1								
32	392	Male Header: **1x6** (Little foot **Thru-Hole** for **Arduino**)	1	Adafruit	https://www.adafru	1	$4.95	1	$4.95	2	$9.90
33		Male Header: **1x8** (Little foot **Thru-Hole** for **Arduino**)	2								
34		Male Header: **1x10** (Little foot **Thru-Hole** for **Arduino**)	1								
35	2822	Male Header: 2x20 (Rasp Pi Thru-Hole for Big Foot)	1	Adafruit	https://www.adafr	1	$0.95	1	$0.95	12	$11.40
36	2976	22 AWG Stranded Core Wire (Button and ESD Wireing Harnesses)	0.16	Adafruit	https://www.adafr	1	$2.95	0.16	$0.47	1.92	$5.66
37	WJ-Z01	Blank Headers 1 x 4	1	Amazon	https://www.amaz	1	$12.98	0.05	$0.65	0.6	$7.79
38		Male Wire Crimps	4								
39	30316-6002HB	Header, 2x8, Ribbon Cable Shrouded (big foot & little foot)	2	Newark	https://www.newa	1	$0.87	2	$1.74	24	$20.86
40		Big Foot PCB	1			1	$6.08	1	$6.08	15	$91.20
41	N/A	Big Foot Stencil	1	PCBWay	https://www.pcbw	1	$14.55	1	$14.55	1	$14.55
42		Little Foot PCB	1			1	$6.13	1	$6.13	15	$91.95
43		Little Foot Stencil	1			1	$14.55	1	$14.55	1	$14.55
								Unit Total:	$90.97	Total:	$758.63

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.	VENDOR	LINK	Ordered in a Pack of	Pack Cost	Qty of Pack Per Fixture	Cost Per Fixture	Order Quantity Required	Total Cost
HARDWARE / FASTENERS											
1	92095A179	BHS M3x0.5 6mm	40	McMaster	https://www.mcma	100	$5.70	0.40	$2.28	4	$22.80
2	92095A181	BHS M3x0.5 8mm	46	McMaster	https://www.mcma	100	$8.90	0.46	$4.09	5	$44.50
3	97654A209	FBHS M3x0.5 25mm	1	McMaster	https://www.mcma	25	$6.00	0.04	$0.24	1	$6.00
4	92125A125	THS M3x0.5 5mm	4	McMaster	https://www.mcma	100	$3.89	0.04	$0.16	1	$3.89
5	92905A602	Set Screw M3x0.5	4	McMaster	https://www.mcma	50	$8.57	0.08	$0.69	1	$8.57
6	4255	Heat-Set Inserts for Plastic	94	Adafruit	https://www.adafr	50	$5.36	1.88	$10.08	19	$101.84
7	92492A711	M2.5x0.45_6mm PHS	4	McMaster	Nylon Pan Head Ph	100	$7.18	0.04	$0.29	1	$7.18
8	36-4687-ND	M2.5x0.45 Nylon Nut	8	Digikey	4687 Keystone Ele	1	$0.25	8.00	$2.00	80	$20.00
9	761-M2107-2545-AL	M2.5x0.45_12mm Aluminum Standoff	4	Mouser	https://www.mous	1	$0.83	4.00	$3.34	40	$33.36
10	761-M2117-2545-AL	M2.5x0.45_22mm Aluminum Standoff	4	Mouser	https://www.mous	1	$0.93	4.00	$3.71	40	$37.12
11	732-10390-ND	M3x0.5_8mm S.S. Standoffs (PCB Mounting Plate)	4	DigiKey	https://www.digike	1	$0.34	4.00	$1.36	40	$13.60
12	89535K16	1/8" Steel Rod 1.0625" Length	2								
13	89535K16	1/8" Steel Rod 1.83" Length	2	McMaster	https://www.mcma	12	$1.78	0.78	$1.39	8	$14.24
14	89535K16	1/8" Steel Rod 0.9" Length	4								
15	98430A116	Push-on Retaining Rings	12	McMaster	https://www.mcma	100	$5.65	0.12	$0.68	2	$11.30
16	9434K23	Compression Springs	2	McMaster	https://www.mcma	5	$4.85	0.40	$1.94	4	$19.40
17	9540K701	Rubber Feet	4	McMaster	https://www.mcma	25	$5.65	0.16	$0.90	2	$11.30
18	NLLZLM320PM25	2.5mm threaded standoffs	4	Amazon	https://www.amaz	1	$12.89	0.10	$1.29	1	$12.89
19	1467A5	Friction Hinge	1	McMaster	https://www.mcma	1	$4.24	1.00	$4.24	10	$42.40
20	1488N603	Closed Foam Push Pads (in inches)	2.23648	McMaster	https://www.mcma	120	$13.13	0.02	$0.24	1	$13.13
21	89015K78	3.38 x 0.74 x 0.08" AL ESD GND	0.1	McMaster	Multipurpose 6061	1	$20.04	0.10	$2.00	1	$20.04
22	8574K132	1/4" x 12" x 12" Sheet Lexane	1	McMaster	https://www.mcma	1	$13.47	1.00	$13.47	10	$134.70
23	9785T803	1/4" x 12" x 12" Sheet HDPE	1	McMaster	Marine-Grade Mois	1	$11.05	1.00	$11.05	10	$110.50
24	5503A37	Ball-End Hex L-Key	1	McMaster	https://www.mcma	1	$0.90	1.00	$0.90	10	$9.00
								Unit Total:	$66.34	Total:	$697.76

Appendix B - Mechanical Drawing

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	MDTA1000	LARGE ENCLOSURE ASM	1
2	MDTA2000	SMALL ENCLOSURE ASM	1
3	9785T803	FIXTURE BOTTOM	1
4	92095A181	M3 x 0.5 8mm BHS	18
5	4255	HEAT-SET INSERTS FOR PLASTIC	8
6	9540K701	UNTHREADED BUMPER	4
7	MDTM0002	HARD STOP BLOCK	2
8	MDTM0003	NAME PLATE	1
9	92095A179	M3 X 0.5 6mm BHS	2
10	PHILLIPS PAN HEAD M2.5X0.45 L 25mm ISO METRIC SCREW	M2.5 PHS	4





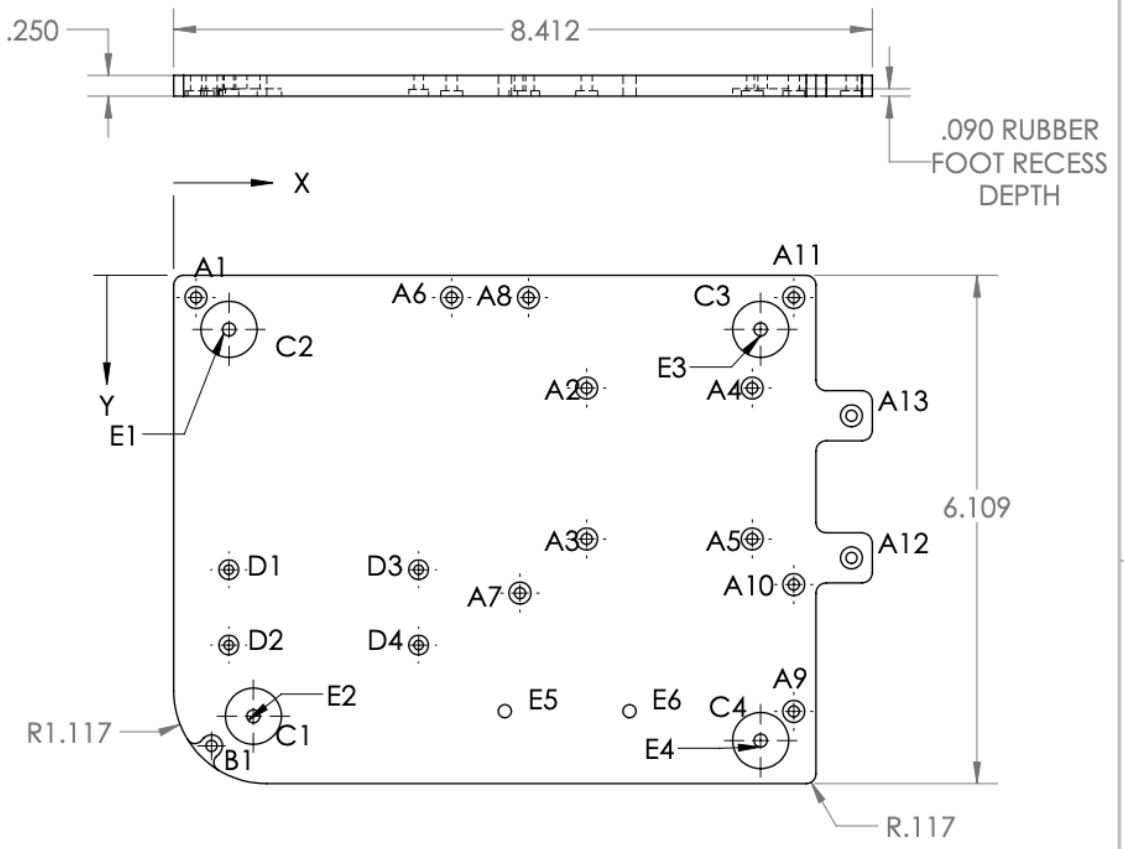
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL -	COMMENTS:		
FINISH -			
DO NOT SCALE DRAWING			

TITLE: TOP ASSEMBLY

SIZE	DWG. NO.	REV
A	MDTA0000	-

SCALE: 1:3 | WEIGHT: | SHEET 1 OF 2

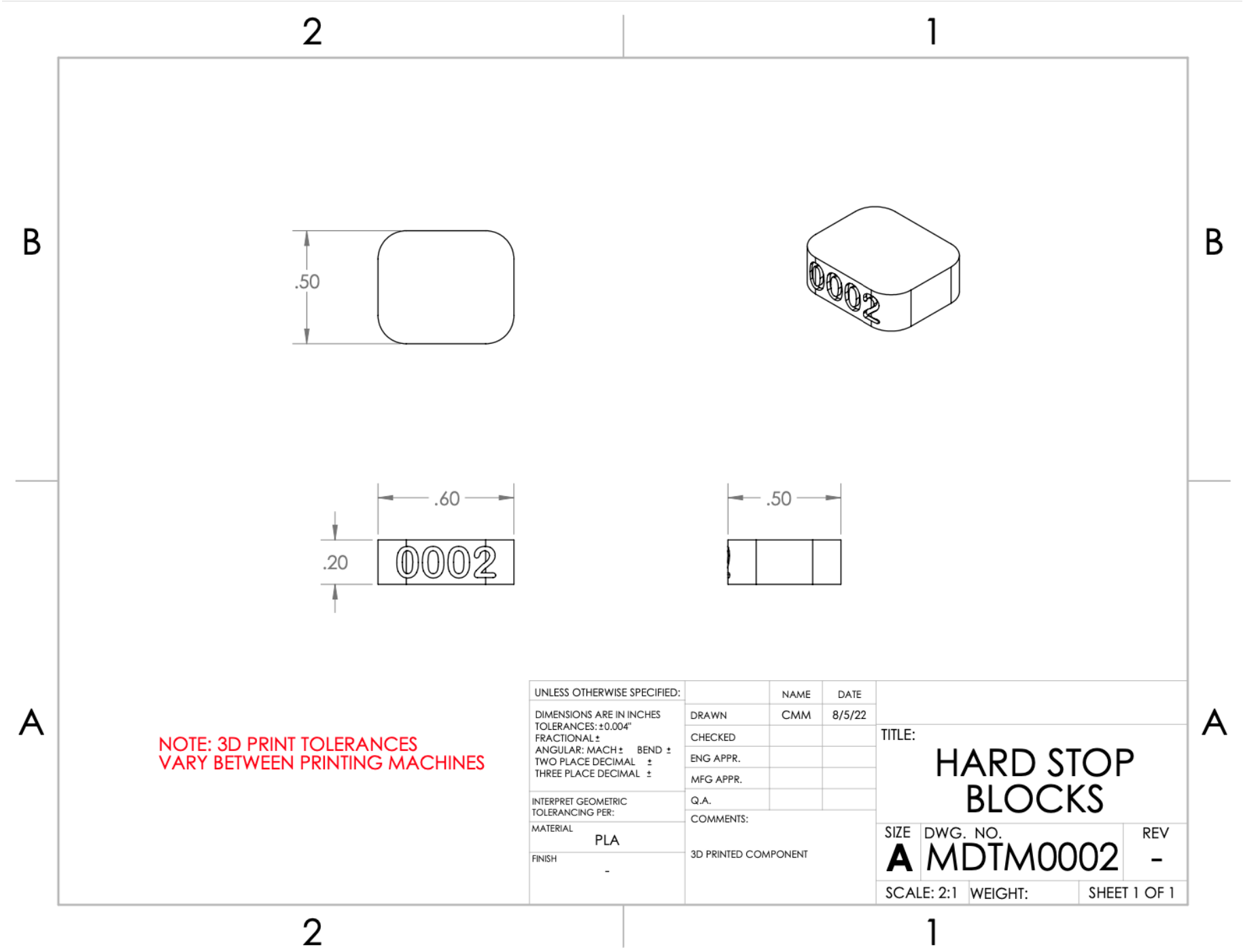
TAG	X LOC	Y LOC	SIZE
A1	.27	.27	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A2	4.97	1.36	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A3	4.97	3.17	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A4	6.97	1.36	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A5	6.97	3.17	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A6	3.34	.27	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A7	4.17	3.82	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A8	4.27	.27	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A9	7.47	5.24	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A10	7.47	3.72	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A11	7.47	.27	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A12	8.16	3.40	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
A13	8.16	1.69	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
B1	.45	5.66	⌀ .134 THRU ALL ⌴ ⌀ .264 ↧ .065
C1	.96	5.30	⌀ .157 THRU ⌴ ⌀ .675 ↧ .090
C2	.67	.65	⌀ .157 THRU ⌴ ⌀ .675 ↧ .090
C3	7.07	.65	⌀ .157 THRU ⌴ ⌀ .675 ↧ .090
C4	7.07	5.59	.675 ↧ .090
D1	.66	3.54	⌀ .114 THRU ALL ⌴ ⌀ .236 ↧ .083
D2	.66	4.44	⌀ .114 THRU ALL ⌴ ⌀ .236 ↧ .083
D3	2.95	3.54	⌀ .114 THRU ALL ⌴ ⌀ .236 ↧ .083
D4	2.95	4.44	⌀ .114 THRU ALL ⌴ ⌀ .236 ↧ .083
E1	.67	.65	⌀ .157 THRU ALL
E2	.96	5.30	⌀ .157 THRU ALL
E3	7.07	.65	⌀ .157 THRU ALL
E4	7.07	5.59	⌀ .157 THRU ALL
E5	3.98	5.24	⌀ .157 THRU ALL
E6	5.48	5.24	⌀ .157 THRU ALL

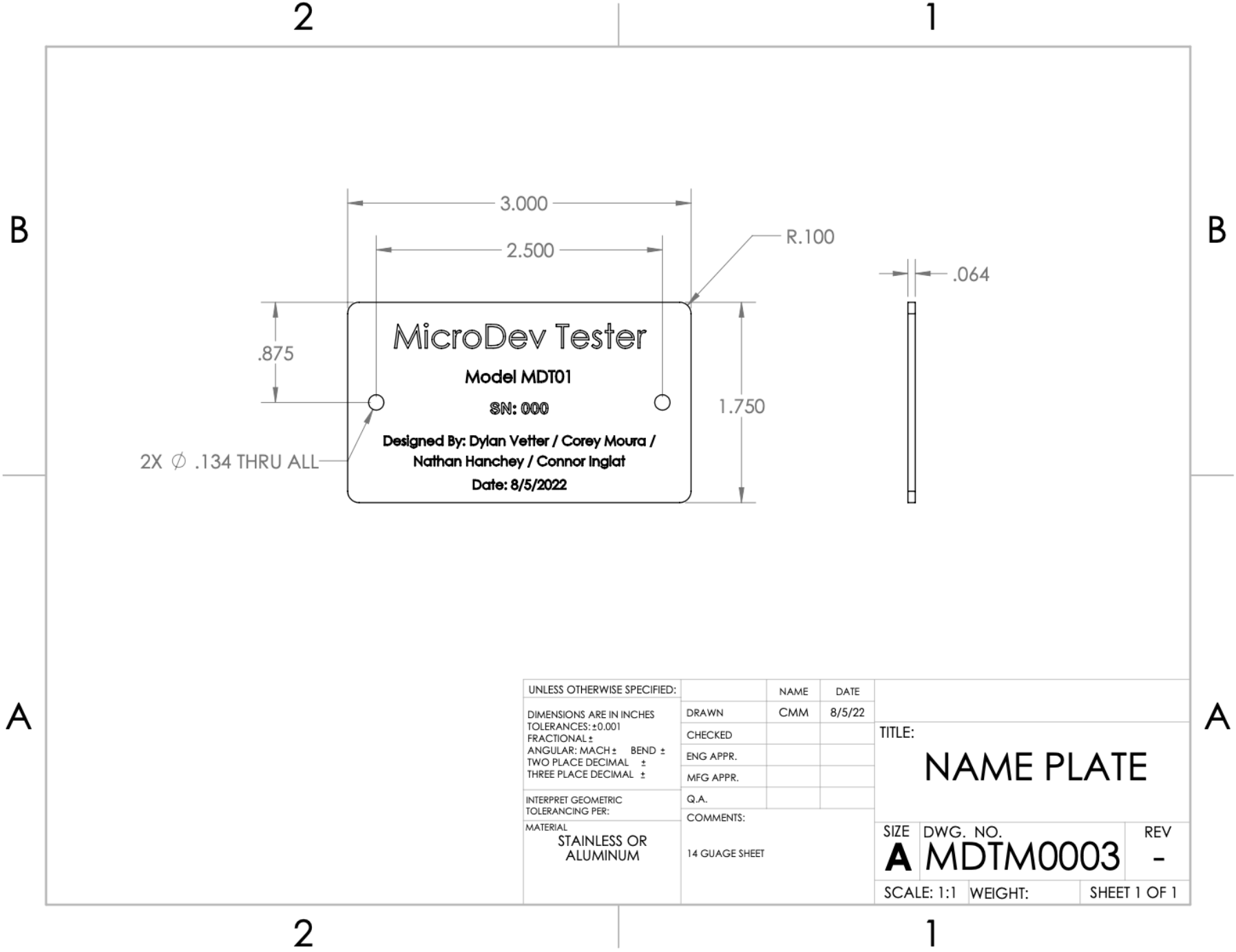


UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.001" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL: PLA	3D PRINTED COMPONENT		
FINISH: -			
DO NOT SCALE DRAWING			

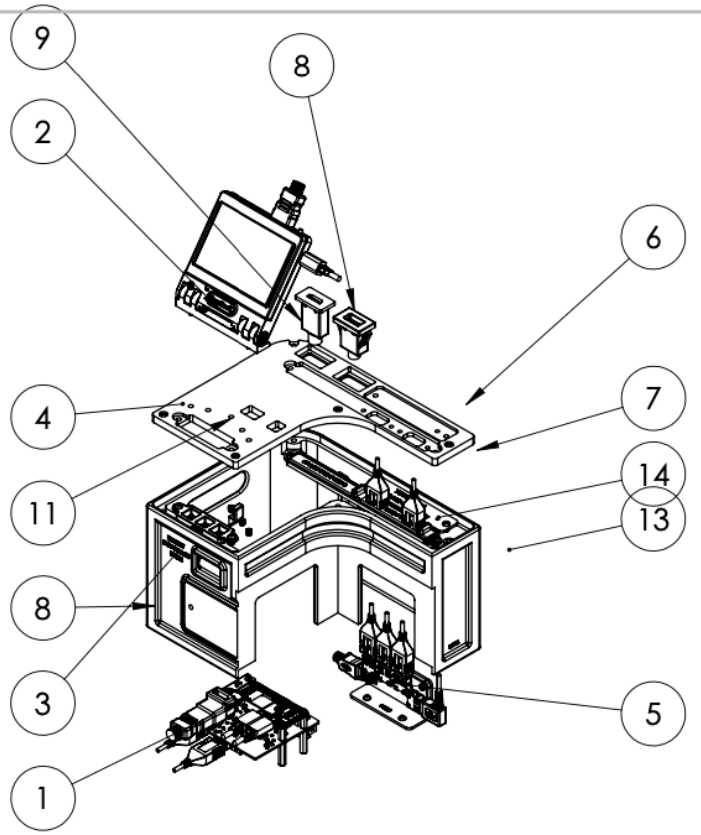
TITLE: **BASE PLATE**

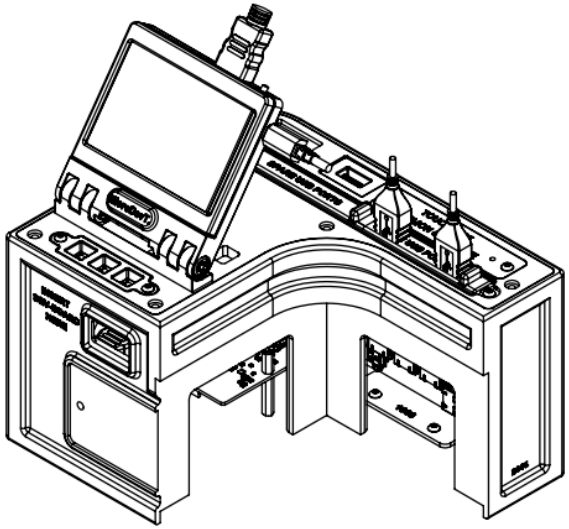
SIZE	DWG. NO.	REV
A	MDTM0001	-
SCALE: 1:2	WEIGHT:	SHEET 1 OF 1


2
1
B
B
.50
0002
.60
.20
0002
.50
A
A
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
HARD STOP BLOCKS
SIZE
A
DWG. NO.
MDTM0002
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1
2
1


3.000
2.500
R.100
.064
.875
1.750
MicroDev Tester
Model MDT01
SN: 000
Designed By: Dylan Vetter / Corey Moura /
Nathan Hanchey / Connor Inglat
Date: 8/5/2022
2X ∅ .134 THRU ALL
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.001
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
STAINLESS OR ALUMINUM
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
14 GUAGE SHEET
TITLE:
NAME PLATE
SIZE
A
DWG. NO.
MDTM0003
REV
-
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1

ITEM NO.	PART NUMBER	DESCRIPTION	MATERIAL	VENDOR	QTY.
1	PI_ZERO_ASM	PI / BIGFOOT PCB ASM	-	-	1
2	MDTA1100	LCD HOUSING ASM	-	IN HOUSE	1
3	MDTM1001	LARGE ENCLOSURE	PLA	IN HOUSE	1
4	8574K132	LARGE ENCLOSURE TOP	12" X 12" X 1/4" CLEAR LEXANE SHEET	MCMASTER CARR	1
5	C1419	USB Hub	-	Chicago Electric	1
6	RB141C1000-114-ND	(B) Power Switch	-	DigiKey	1
7	4052	(C) Snap Mount USB-C	-	Adafruit	1
8	4055	(G) Snap Mount USB-A	-	Adafruit	2
9	4054	(I) Snap Mount HDMI	-	Adafruit	1
10	92095A179	M3 X 0.5 6mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	12
11	92095A181	M3 X 0.5 8mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	7
12	4255	HEAT-SET INSERTS FOR PLASTIC	BRASS	ADAFRUIT	30
13	MDTM1004	STORAGE POCKET	PLA	IN HOUSE	1
14	89015K78	3.38 X 0.74 X 0.08" AL ESD GND	12 GUAGE ALUMINUM SHEET	MCMASTER CARR	1
15	MDTA1200				1
16	MDTA1300				1
17	2197	(J) HDMI FLAT CABLE	-	ADAFRUIT	1
18	4298	(A) PI POWER SUPPLY USB-C	-	ADAFRUIT	1
19	69405K5	(Q) TERMINAL FOR ESD GND PLATE	-	MCMASTER CARR	1
20	7849K31	(S) POWER SWITCH TERMINALS		MCMASTER CARR	3





	NAME	DATE
DRAWN	CMM	8/5/22
CHECKED		
ENG APPR.		
MFG APPR.		
Q.A.		

COMMENTS:

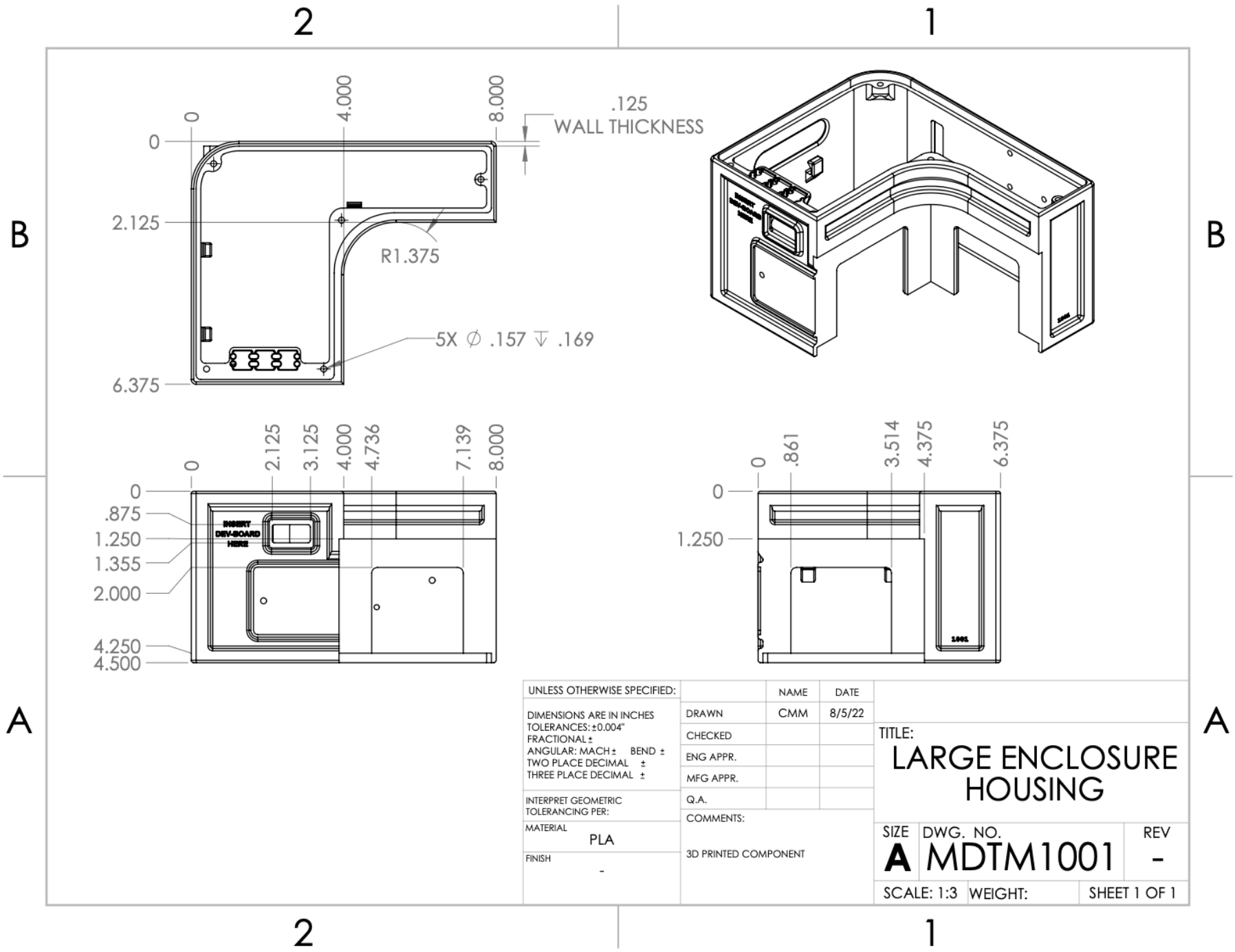
LARGE ENCLOSURE ASSEMBLY

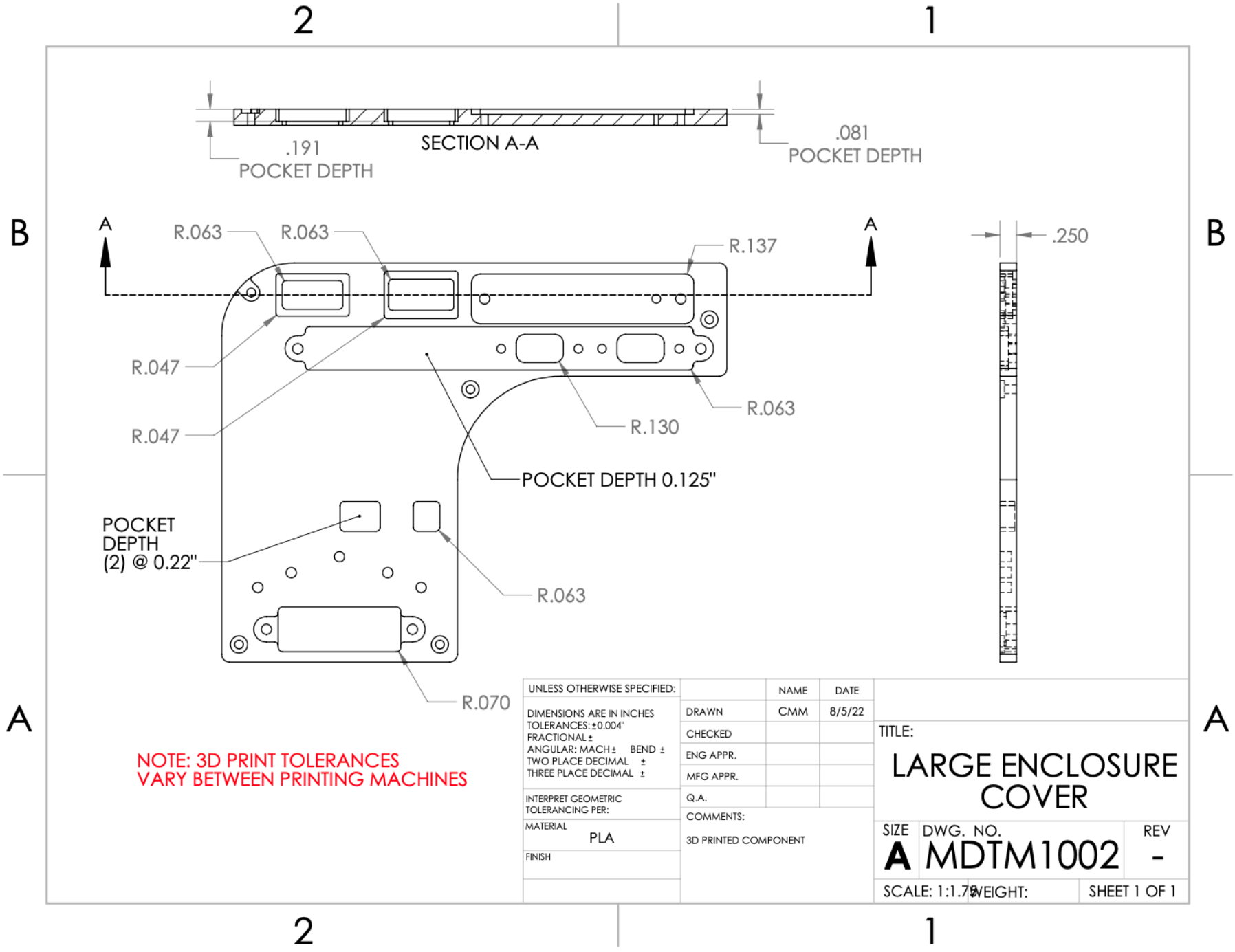
TITLE:

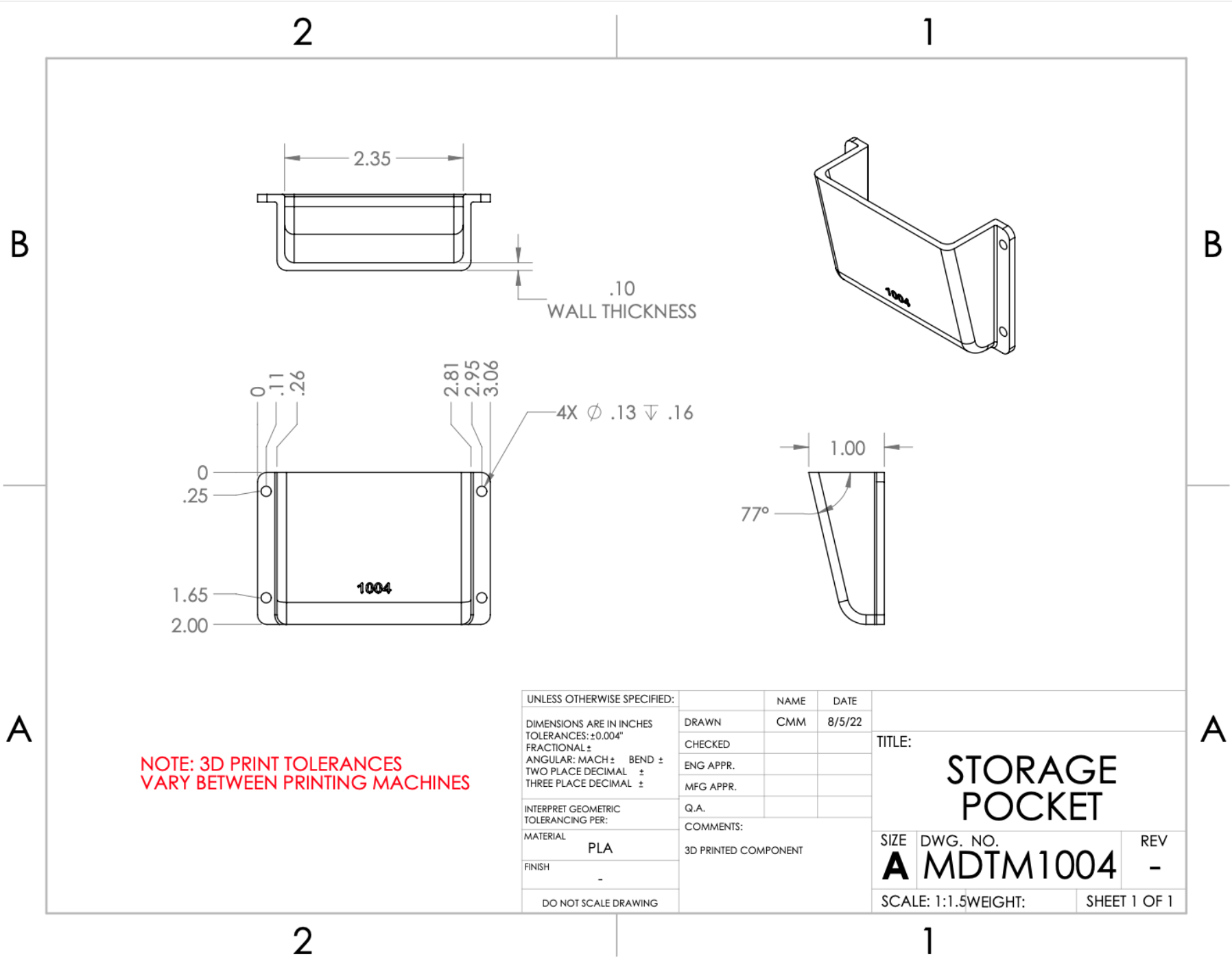
LARGE ENCLOSURE ASM

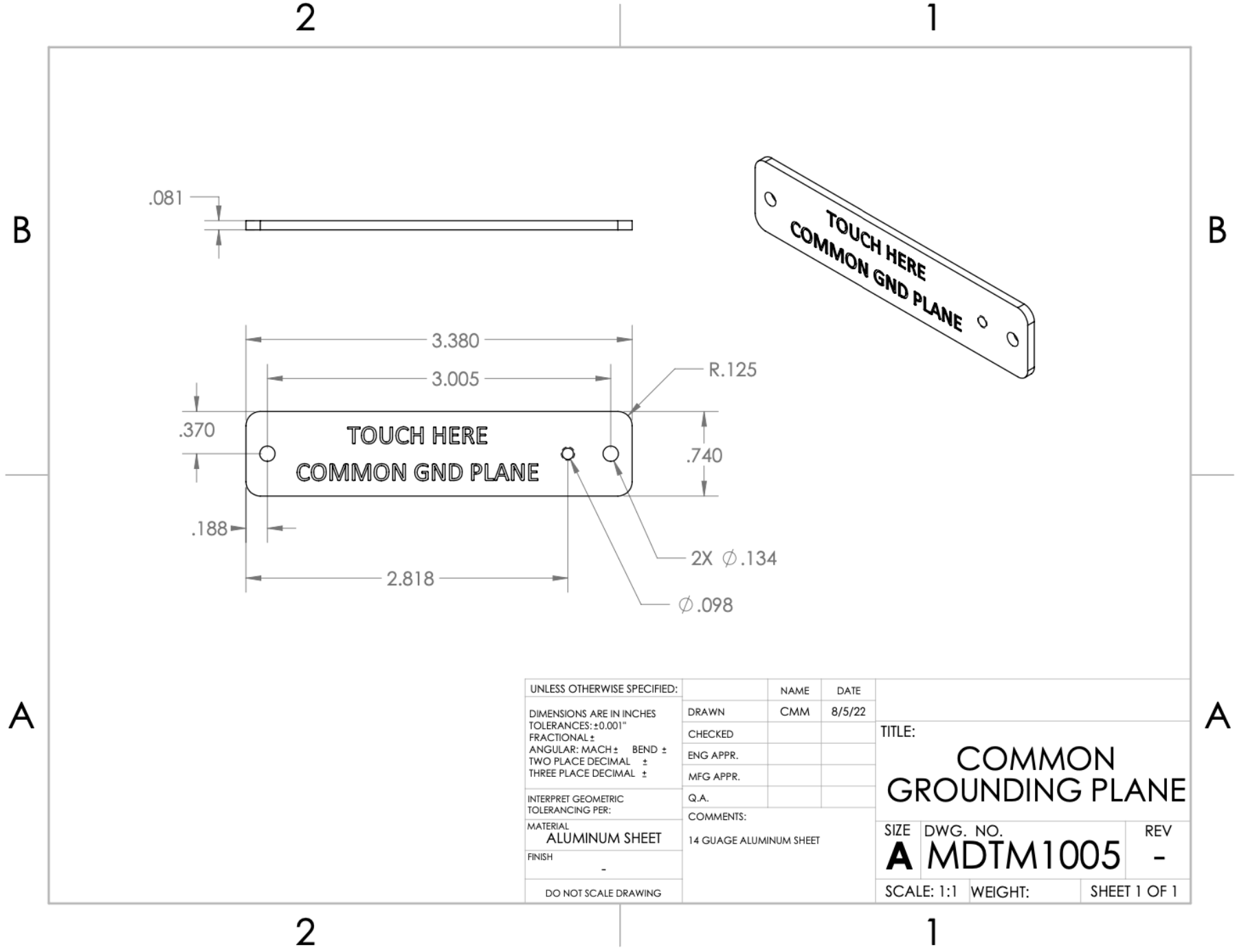
SIZE	DWG. NO.	REV
A	MDTA1000	-

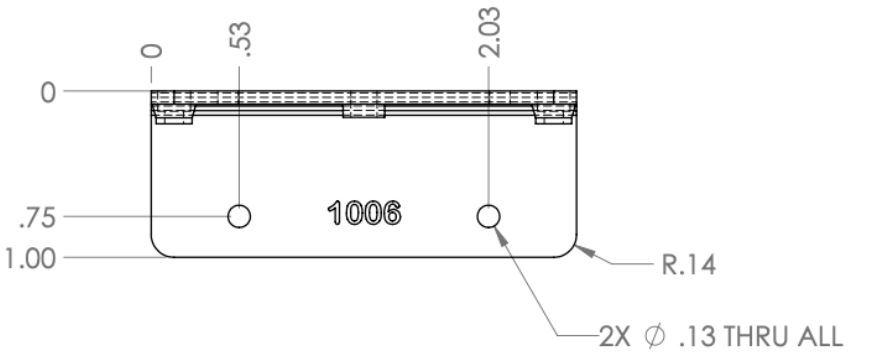
SCALE: 1:5 | WEIGHT: | SHEET 1 OF 2

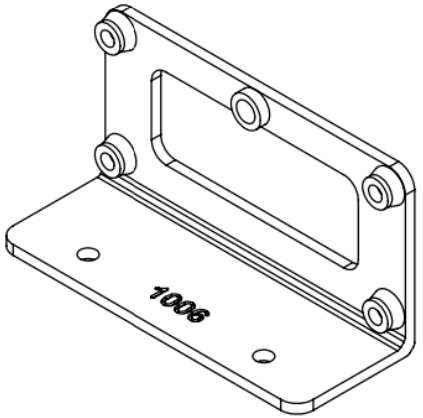

2
1
B
A
0
4.000
8.000
.125
WALL THICKNESS
2.125
R1.375
5X ⌀ .157 ↧ .169
6.375
0
2.125
3.125
4.000
4.736
7.139
8.000
.875
1.250
1.355
2.000
4.250
4.500
INSERT DEV-BOARD HERE
0
.861
3.514
4.375
6.375
1.250
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LARGE ENCLOSURE HOUSING
SIZE
A
DWG. NO.
MDTM1001
REV
-
SCALE: 1:3
WEIGHT:
SHEET 1 OF 1

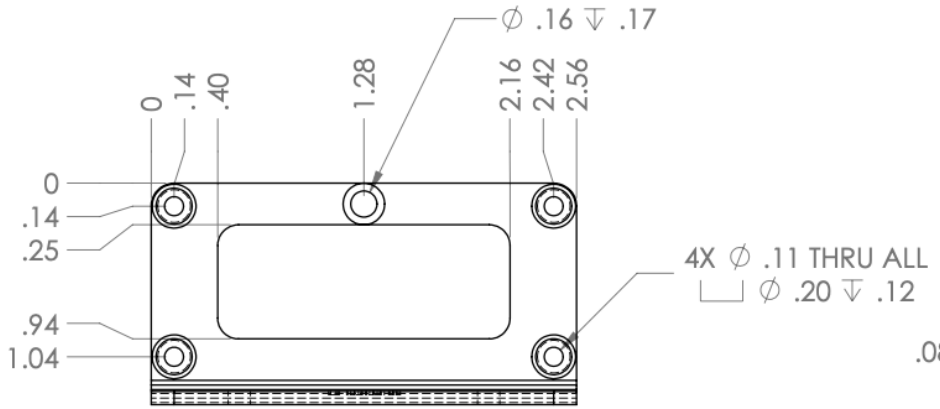

2
1
B
A
SECTION A-A
.191
POCKET DEPTH
.081
POCKET DEPTH
R.063
R.063
R.137
R.047
R.047
R.130
R.063
POCKET DEPTH 0.125"
POCKET
DEPTH
(2) @ 0.22"
R.063
R.070
.250
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LARGE ENCLOSURE COVER
SIZE
A
DWG. NO.
MDTM1002
REV
-
SCALE: 1:1.75
WEIGHT:
SHEET 1 OF 1

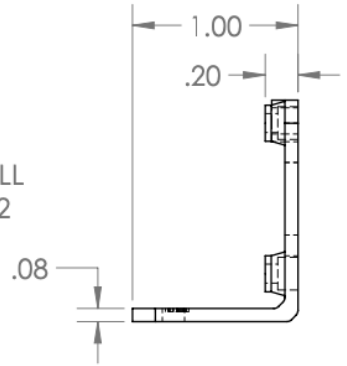

2
1
B
A
2.35
.10
WALL THICKNESS
1004
0
.11
.26
2.81
2.95
3.06
4X ∅ .13 ↧ .16
0
.25
1.65
2.00
1004
1.00
77°
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
STORAGE POCKET
SIZE
A
DWG. NO.
MDTM1004
REV
-
SCALE: 1:1.5
WEIGHT:
SHEET 1 OF 1


2
1
B
A
.081
3.380
3.005
R.125
.370
TOUCH HERE
COMMON GND PLANE
.740
.188
2X ∅.134
2.818
∅.098
TOUCH HERE
COMMON GND PLANE
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.001"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
ALUMINUM SHEET
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
14 GUAGE ALUMINUM SHEET
TITLE:
COMMON GROUNDING PLANE
SIZE
A
DWG. NO.
MDTM1005
REV
-
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1







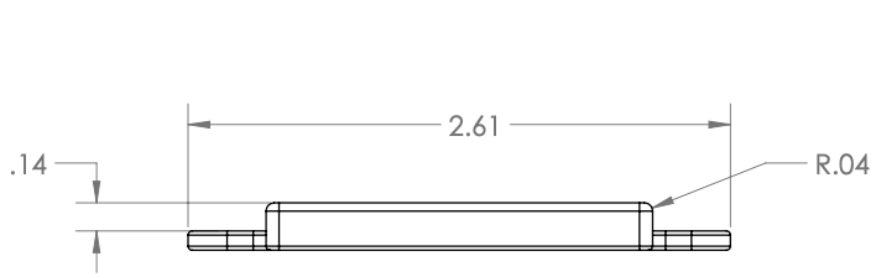


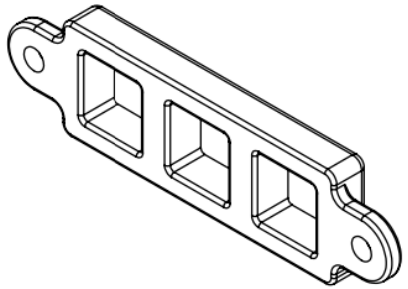
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES

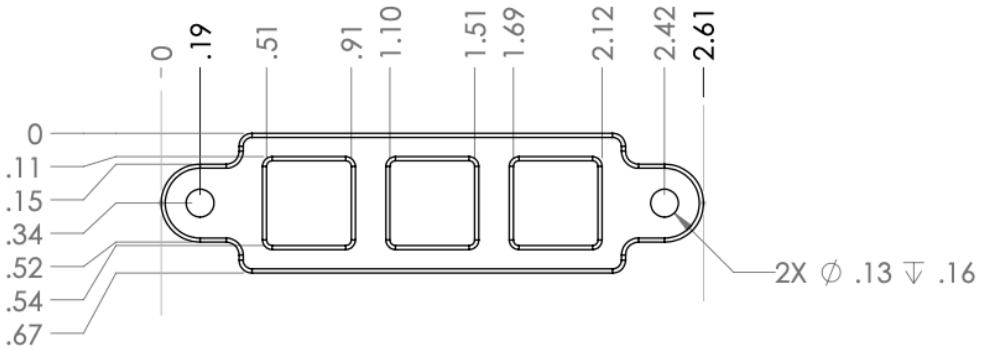
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ± 0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL: PLA	3D PRINTED COMPONENT		
FINISH: -			

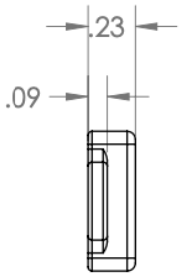
TITLE: USB HUB MOUNTING BRACKET

SIZE	DWG. NO.	REV
A	MDTM1006	-
SCALE: 1:1	WEIGHT:	SHEET 1 OF 1









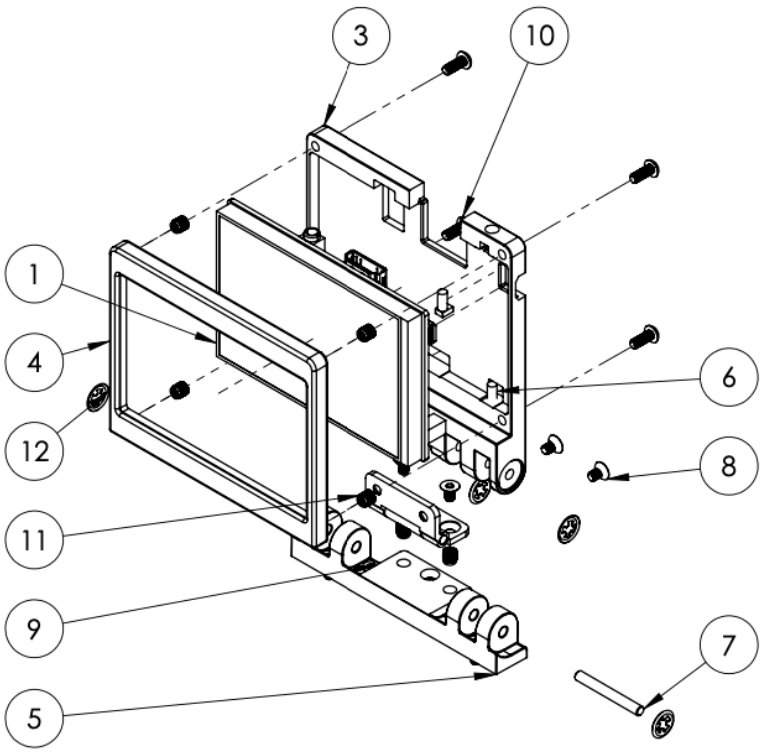
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES

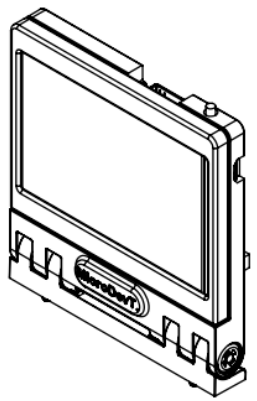
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL CARBON FIBER NYLON	3D PRINTED COMONENT		
FINISH -			

TITLE:
BUTTON PANEL COVER

SIZE	DWG. NO.	REV
A	MDTM1008	-
SCALE: 1.25	WEIGHT:	SHEET 1 OF 1

ITEM NO.	PART NUMBER	DESCRIPTION	Material	Vendor	QTY.
1	Pi4-2016011800	(O) 3.5" LCD	-	Amazon	1
2	FE-USB-ACB590-6IN	(K) Micro to USB-A (LCD to Snap USB-A)	-	Adafruit	1
3	MDTM1102	LCD Housing	Carbon FIber Nylon	-	1
4	MDTM1103	LCD Retainer Bezel	Carbon Fiber Nylon	-	1
5	MDTM1101	LCD Housing Hinge Base	Carbon Fiber Nylon	-	1
6	MDTM1104	LCD Power Button Extension	PLA	-	1
7	89535K16	1/8" Steel Rod 1.0625" Length	Stainless Steel	McMaster Carr	2
8	92125A125	M3x0.5 5.5mm FTHS	Stainless Steel	McMaster Carr	4
9	92095A179	M3 x 0.5 6mm BHS	18-8 Stainless Steel	McMaster-Carr	5
10	92095A181	M3 x 0.5 8mm BHS	18-8 Stainless Steel	McMaster-Carr	4
11	4255	Heat-Set Inserts for Plastic	Brass	Adafruit	8
12	98430A116	Push-on External Retaining Rings	1060-1090 Spring Steel	McMaster-Carr Supply Company	4

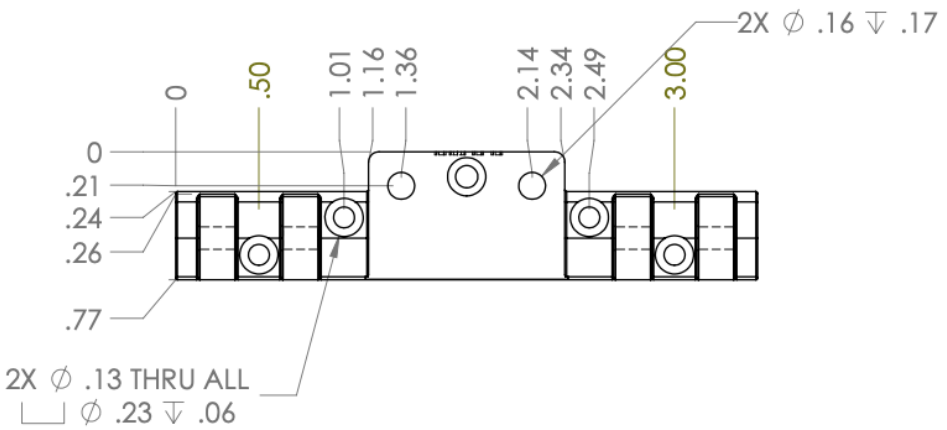


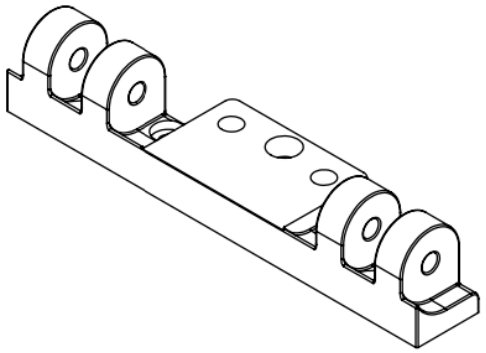


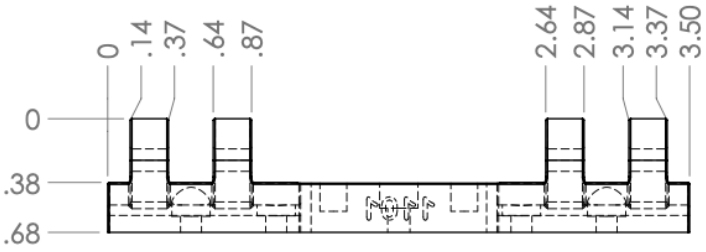
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL -	COMMENTS:		
FINISH - DO NOT SCALE DRAWING			

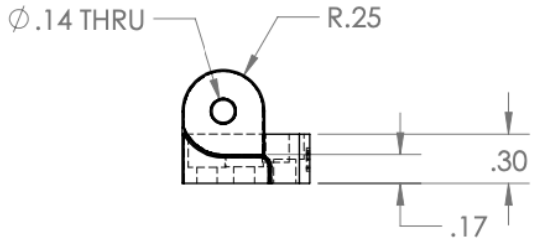
TITLE:
LCD ASSEMBLY

SIZE	DWG. NO.	REV
A	MDTA1100	-
SCALE: 1:2	WEIGHT:	SHEET 1 OF 1







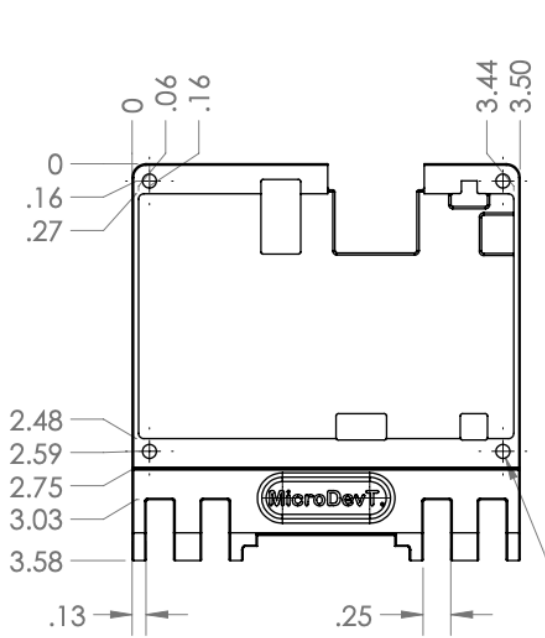


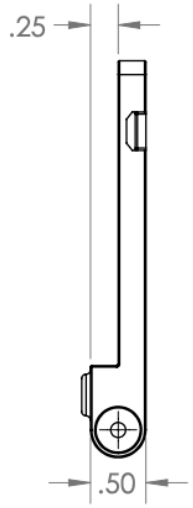
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN MACHINES

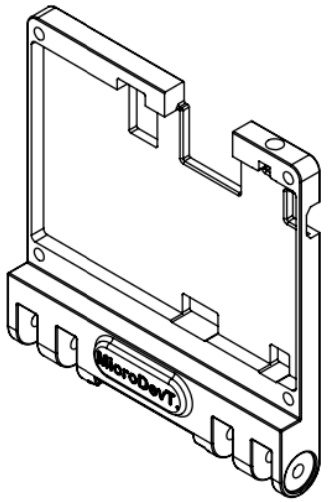
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS: 3D PRINTED COMPONENT		
MATERIAL CARBON FIBER NYLON			
FINISH -			
DO NOT SCALE DRAWING			

TITLE:
LCD HOUSING BRACKET

SIZE	DWG. NO.	REV
A	MDTM1101	-
SCALE: 1:1	WEIGHT:	SHEET 1 OF 1







4X ∅ .13 THRU ALL
⌴ ∅ .23 ↧ .06

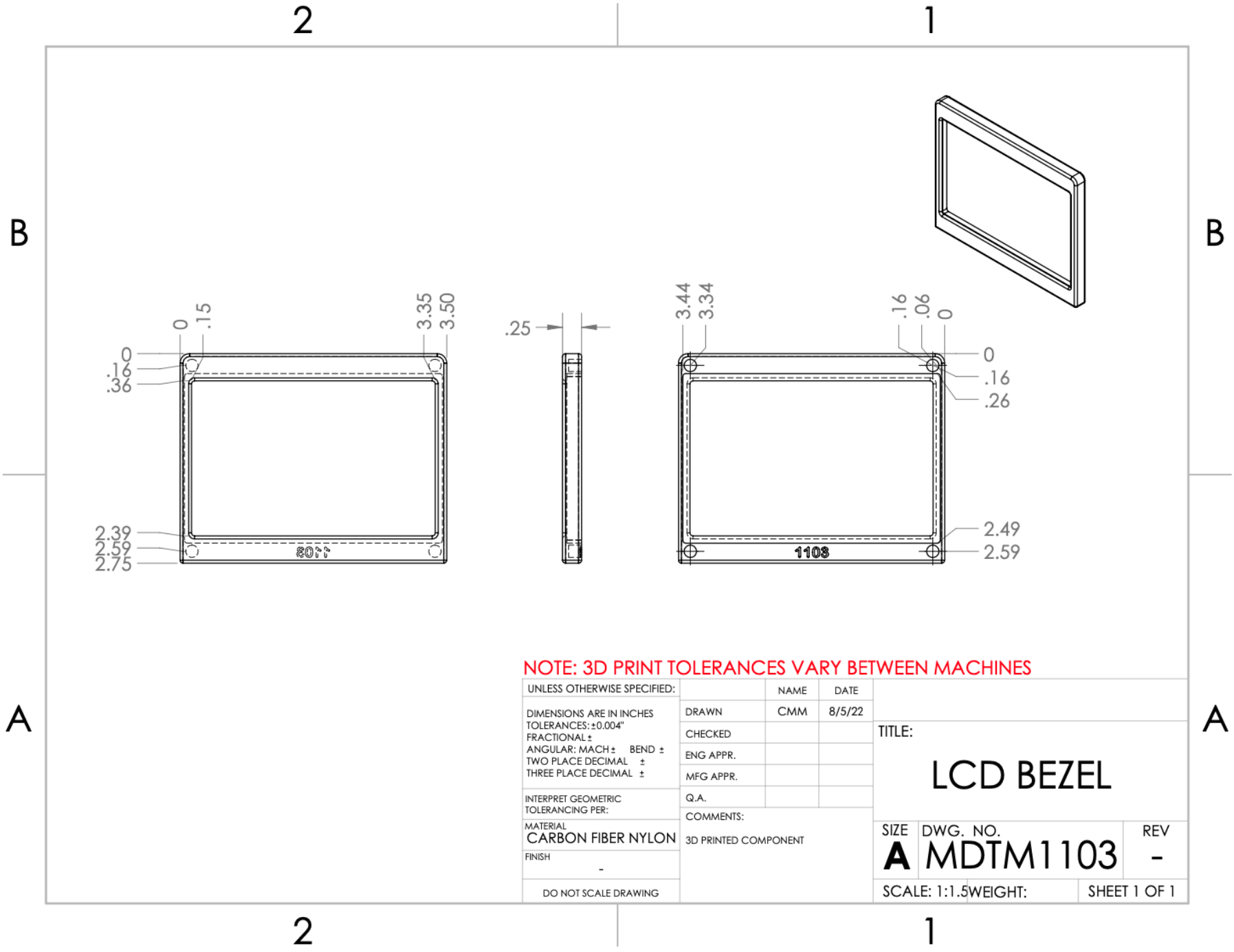
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES

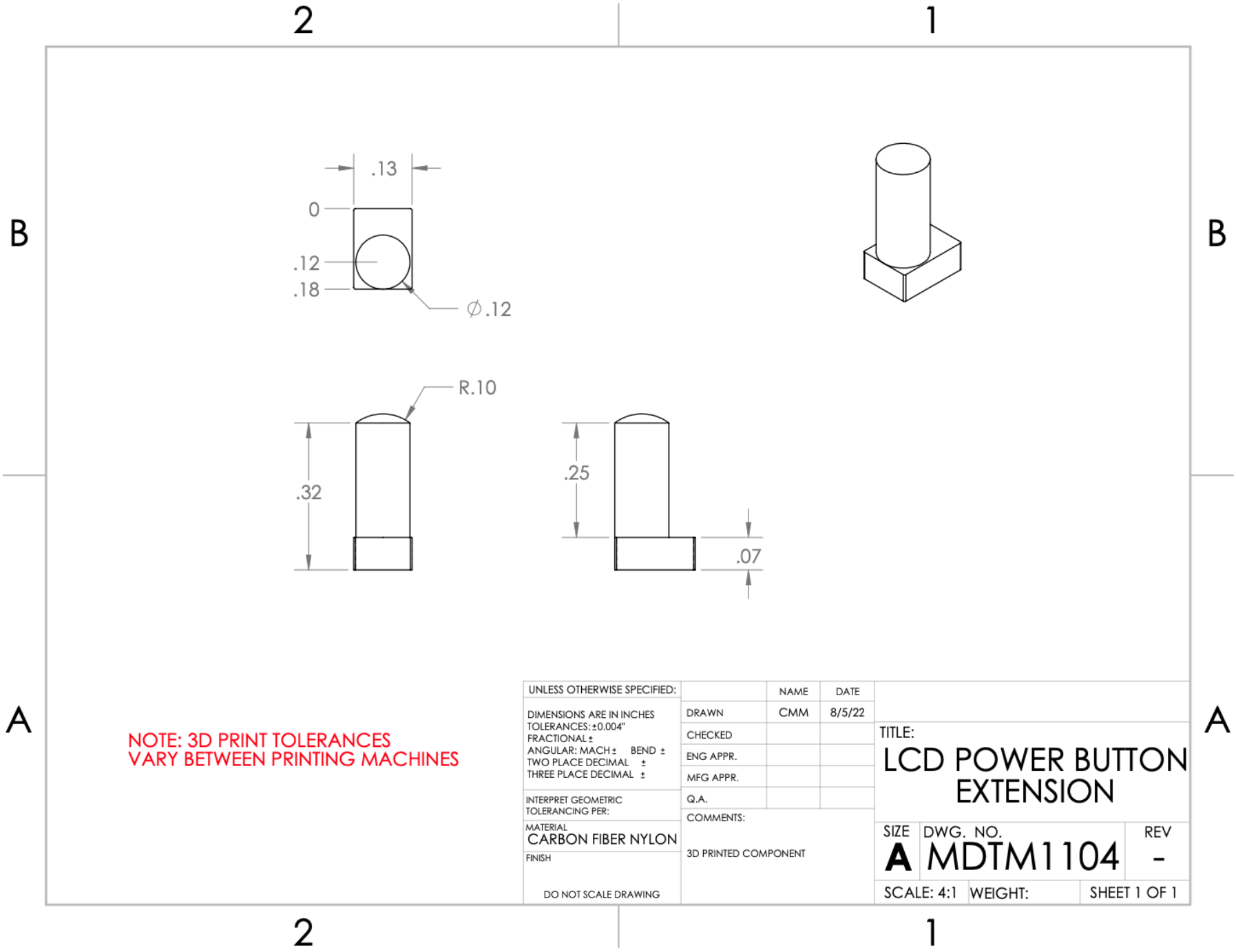
		NAME	DATE
UNLESS OTHERWISE SPECIFIED:			
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL CARBON FIBER NYLON	3D PRINTED COMPONENT		
FINISH -			
DO NOT SCALE DRAWING			

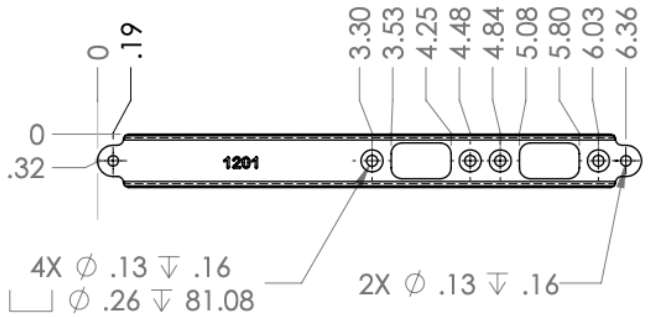
TITLE:

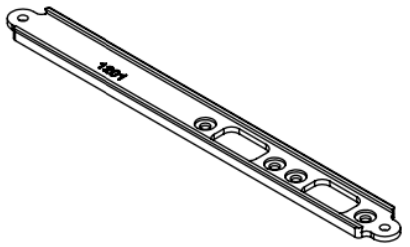
LCD HOUSING

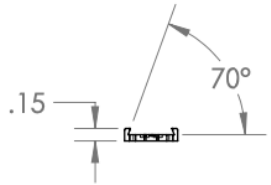
SIZE	DWG. NO.	REV
A	MDTM1102	-
SCALE: 1:1.5	WEIGHT:	SHEET 1 OF 1


2
1
B
A
0
.15
3.35
3.50
0
.16
.36
2.39
2.59
2.75
.25
3.44
3.34
.16
.06
0
0
.16
.26
2.49
2.59
1103
NOTE: 3D PRINT TOLERANCES VARY BETWEEN MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LCD BEZEL
SIZE
A
DWG. NO.
MDTM1103
REV
-
SCALE: 1:1.5
WEIGHT:
SHEET 1 OF 1


2
1
B
A
.13
0
.12
.18
∅.12
R.10
.32
.25
.07
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LCD POWER BUTTON EXTENSION
SIZE
A
DWG. NO.
MDTM1104
REV
-
SCALE: 4:1
WEIGHT:
SHEET 1 OF 1







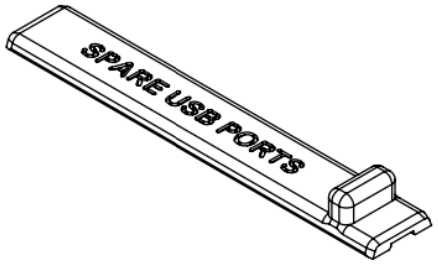
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES

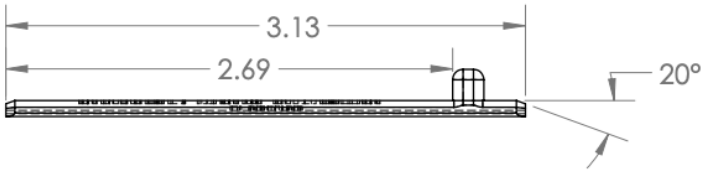
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL CARBON FIBER NYLON	3D PRINTED COMPONENT		
FINISH -			

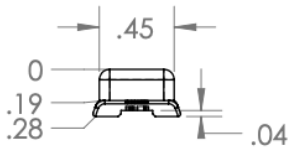
TITLE:
SPARE USB COVER HOUSING

SIZE	DWG. NO.	REV
A	MDTM1201	-

SCALE: 1:2 | WEIGHT: | SHEET 1 OF 1







NOTE: 3D PRINT TOLERANCES VARY BETWEEN PRINTING MACHINES

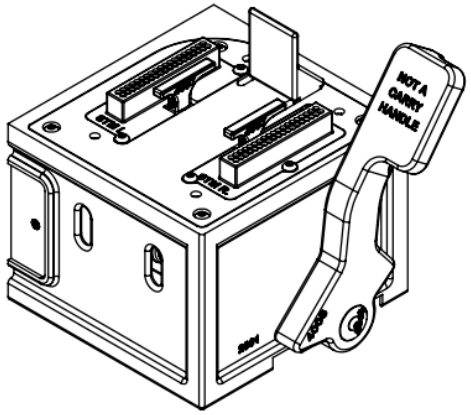
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ± 0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL CARBON FIBER NYLON	COMMENTS: 3D PRINTED COMPONENT		
FINISH			

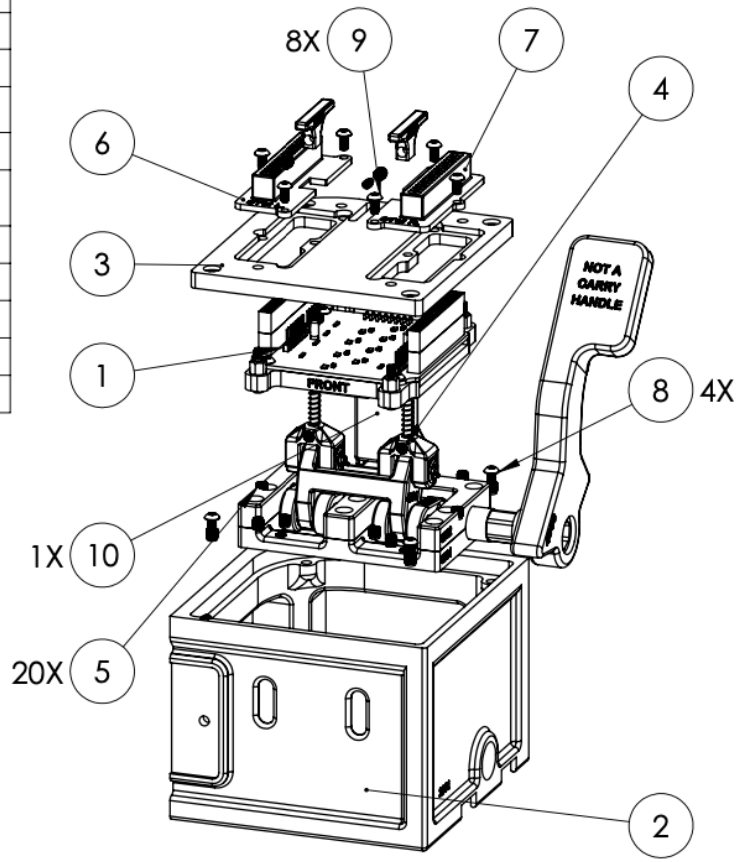
TITLE: SPARE USB COVER

SIZE	DWG. NO.	REV
A	MDTM1202	-

SCALE: 1:1 | WEIGHT: | SHEET 1 OF 1

ITEM NO.	PART NUMBER	DESCRIPTION	MATERIAL	VENDOR	QTY.
1	MDTA3000	SMALL ENCLOSURE ASM	-	-	1
2	MDTM2001	SMALL ENCLOSURE	PLA	IN HOUSE	1
3	8574K132	SMALL ENCLOSURE TOP	12" X 12" X 1/4" CLEAR LEXANE SHEET	MCMASTER CARR	1
4	MDTA4000	EJECTION CRANK ASM	-	-	1
5	4255	HEAT-SET INSERTS FOR PLASTIC	BRASS	ADAFRUIT	20
6	MDTM2003_STM.SLDPRT	STM LEFT HEADER COVER	CARBON FIBER NYLON	-	1
7	MDTM2004	RIGHT HEADER COVER	N/A	IN HOUSE	1
8	92095A181	M3 X 0.5 8mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	4
9	92095A179	M3 X 0.5 6mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	8
10	MDTM2005	SLIDING ALIGNMENT POST	CARBON FIBER NYLON	-	1





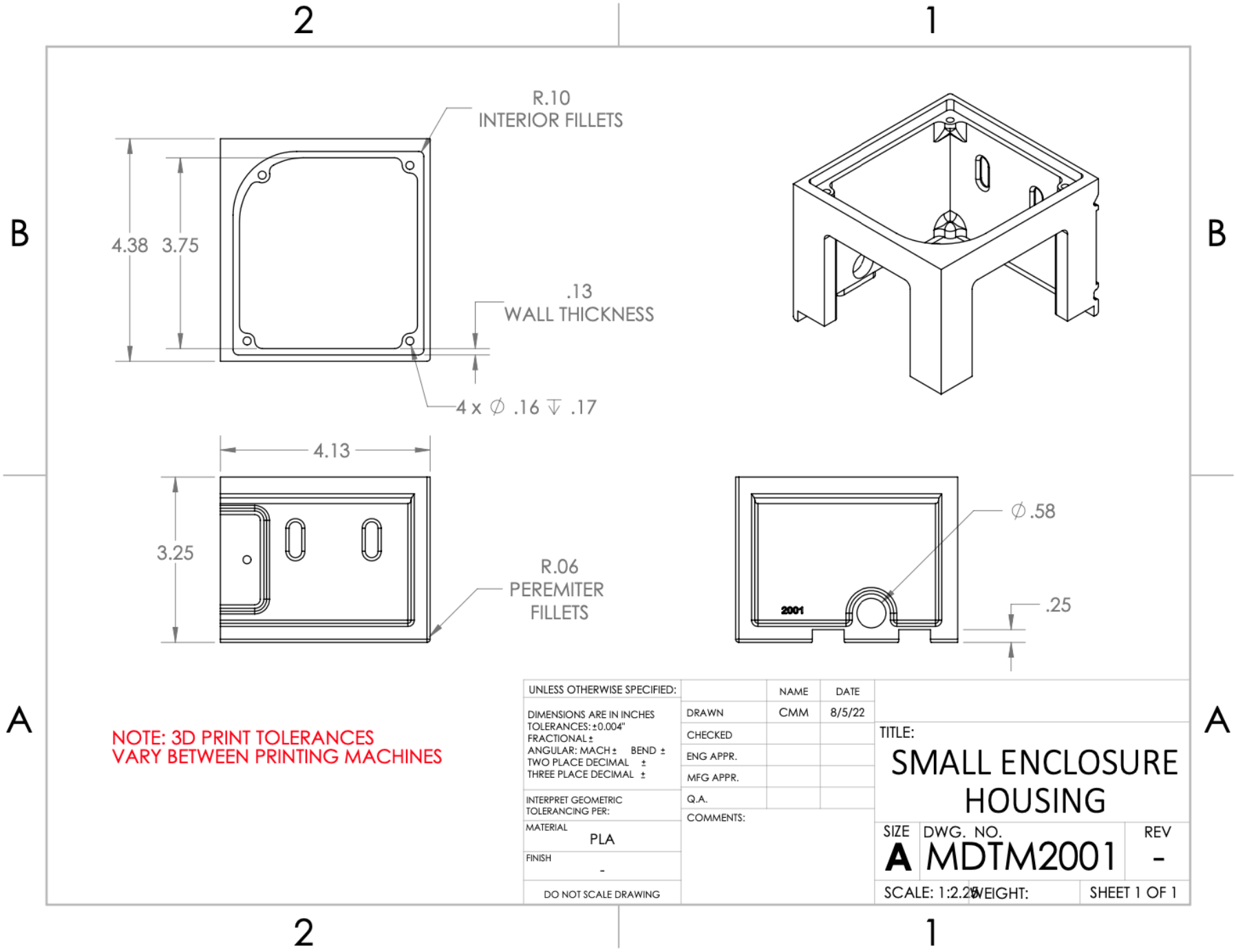
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS: SMALL ENCLOSURE ASSEMBLY		
MATERIAL -			
FINISH -			
DO NOT SCALE DRAWING			

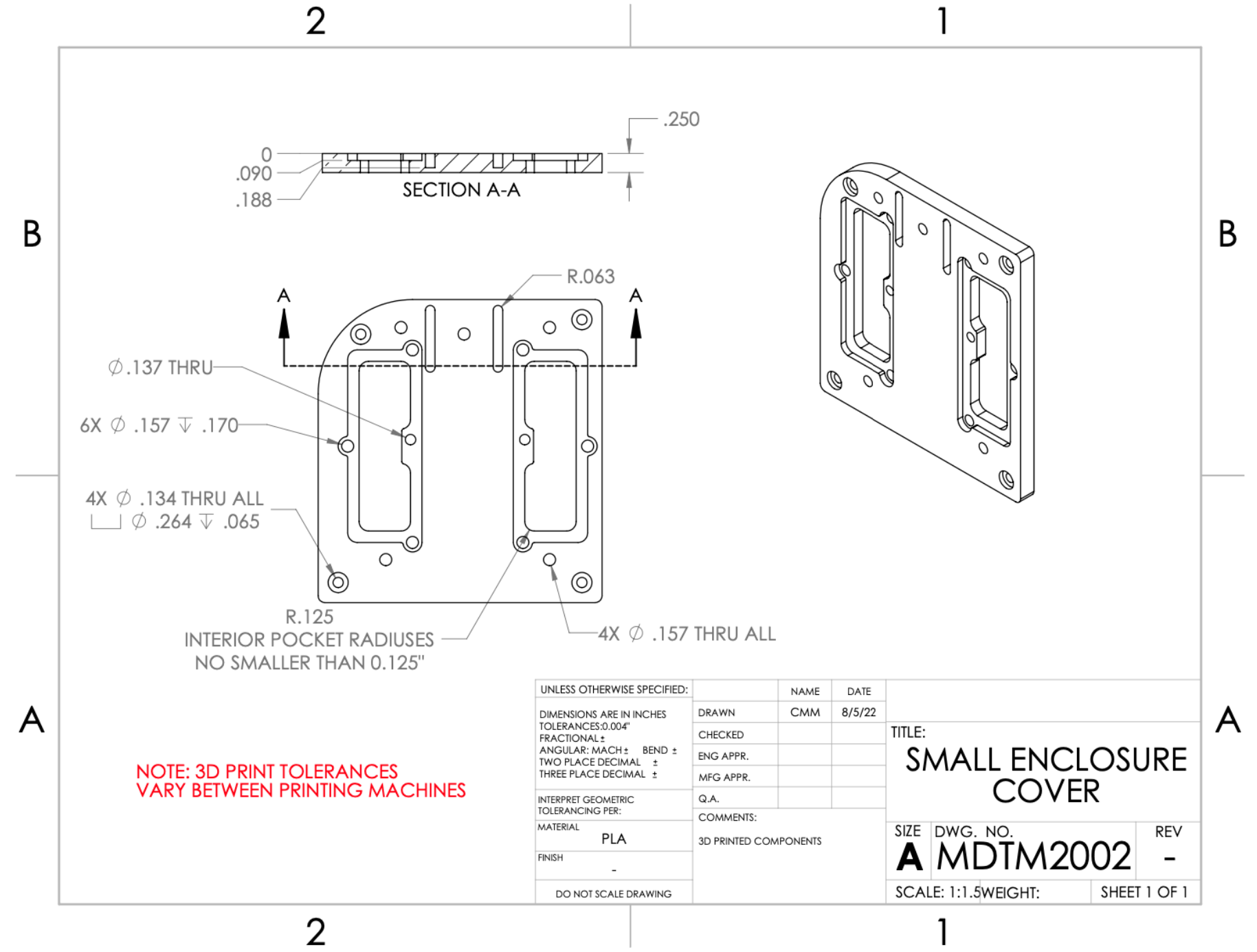
TITLE:

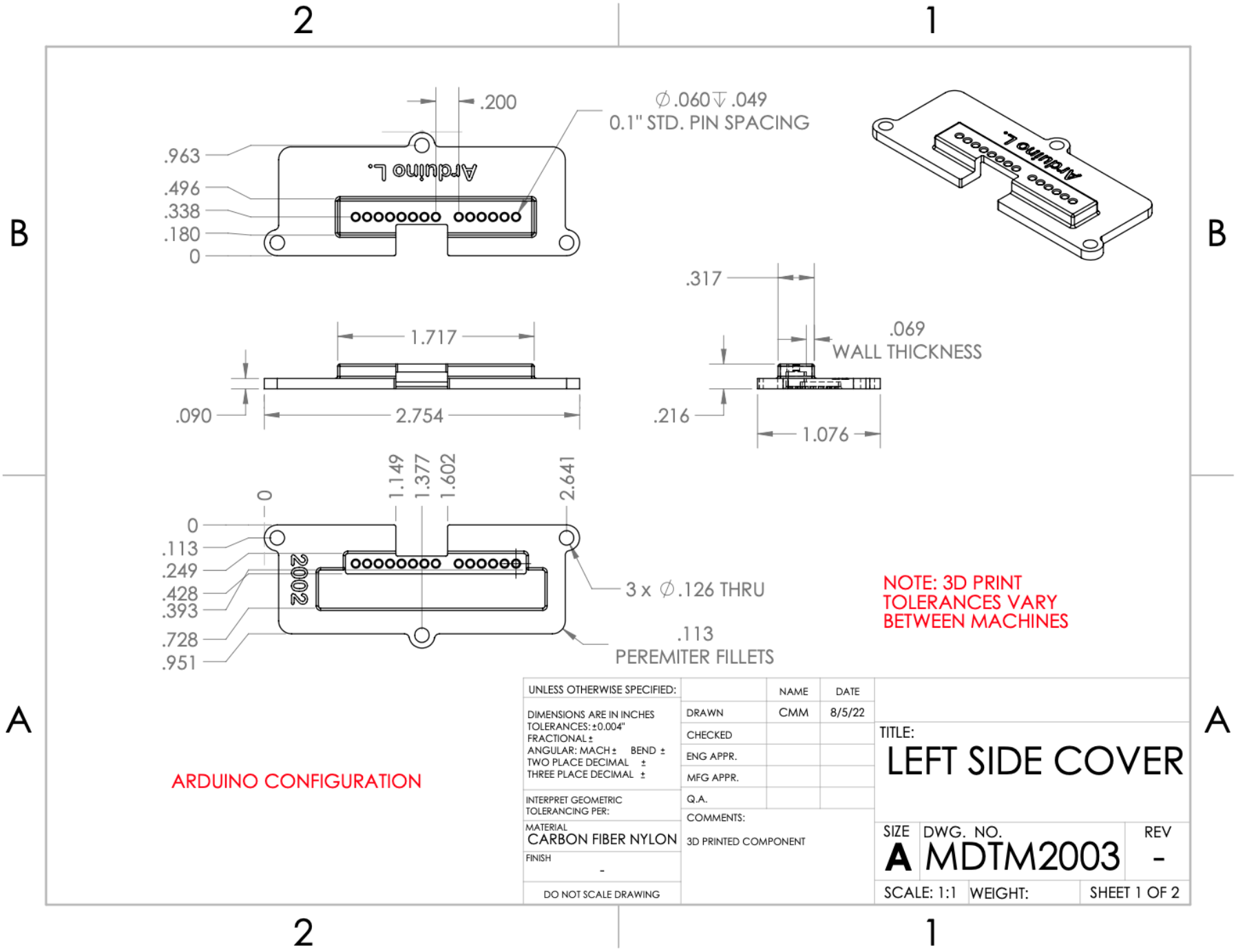
SMALL ENCLOSURE ASM

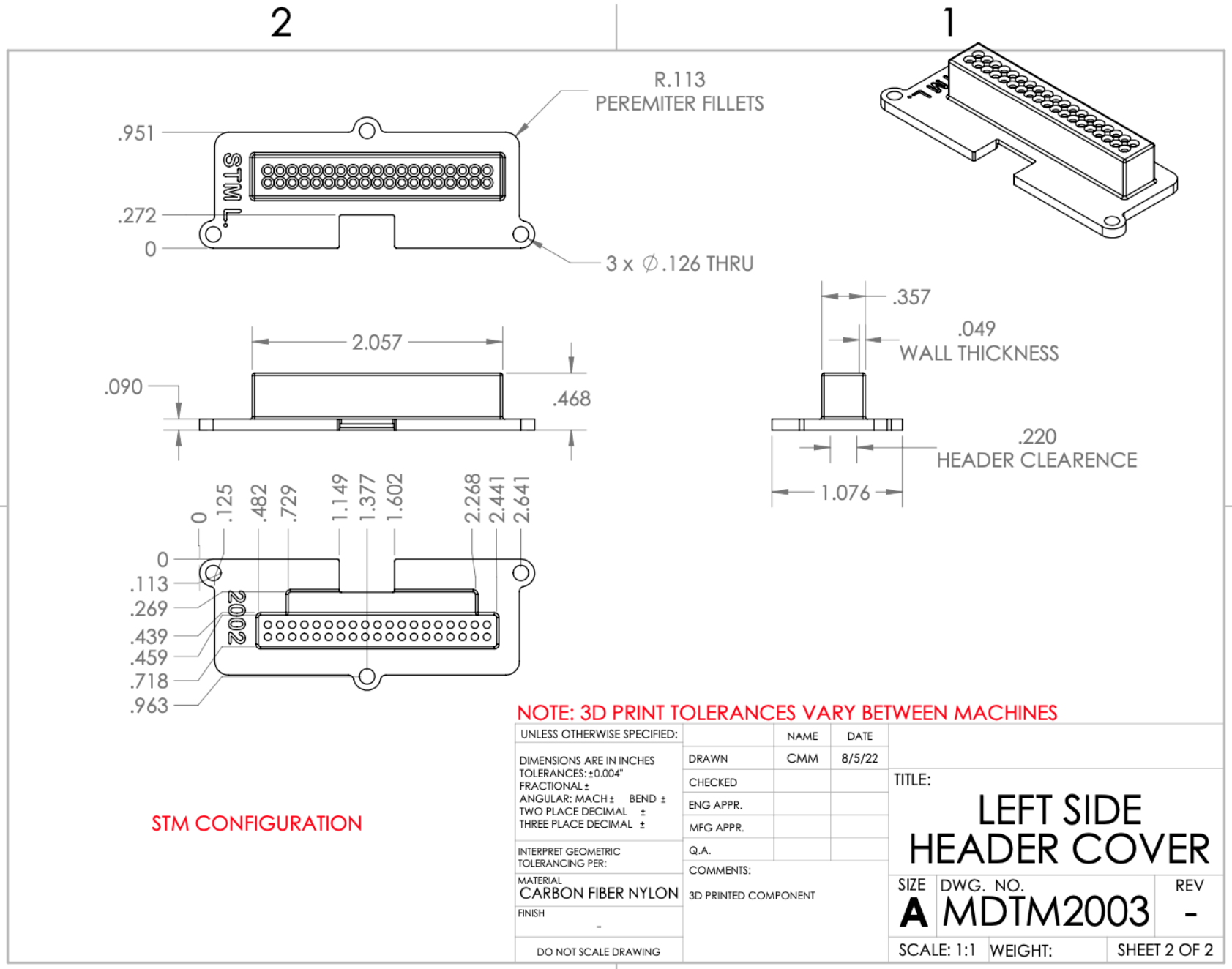
SIZE	DWG. NO.	REV
A	MDTA2000	-

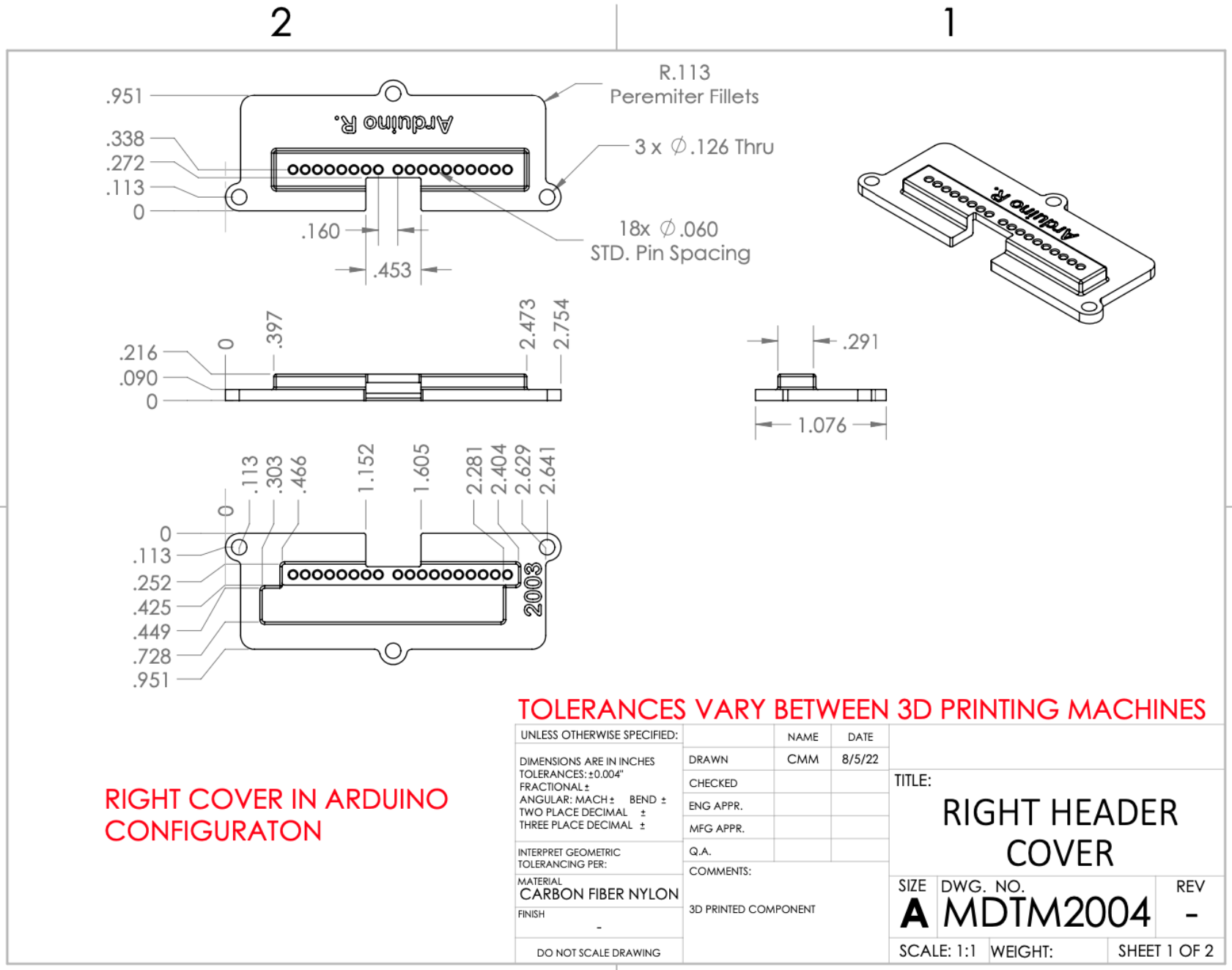
SCALE: 1:2.5 WEIGHT: SHEET 1 OF 1

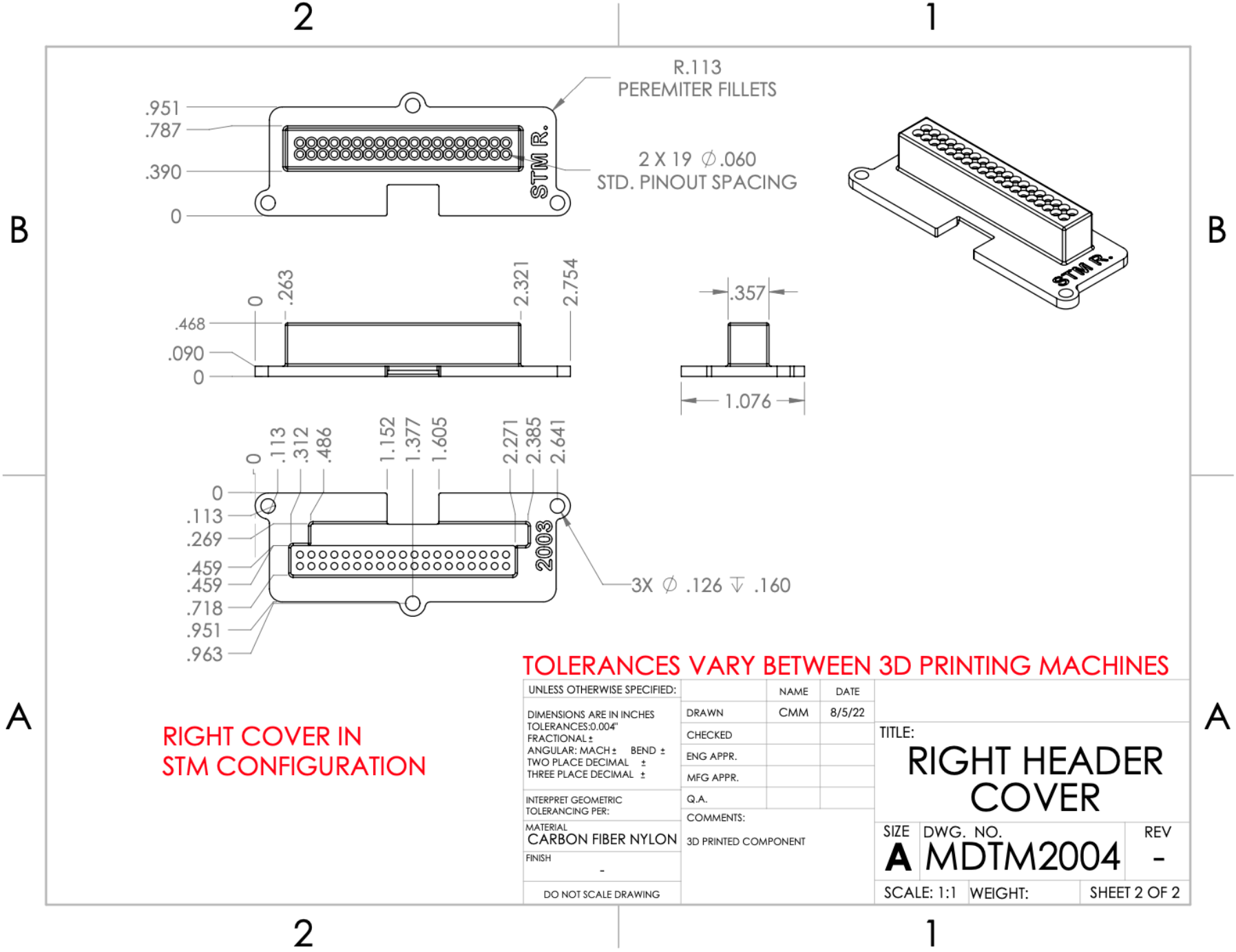

2
1
B
A
R.10
INTERIOR FILLETS
4.38
3.75
.13
WALL THICKNESS
4 x ∅ .16 ↧ .17
4.13
3.25
R.06
PEREMITER
FILLETS
∅.58
.25
2001
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SMALL ENCLOSURE HOUSING
SIZE
A
DWG. NO.
MDTM2001
REV
-
SCALE: 1:2.25
WEIGHT:
SHEET 1 OF 1

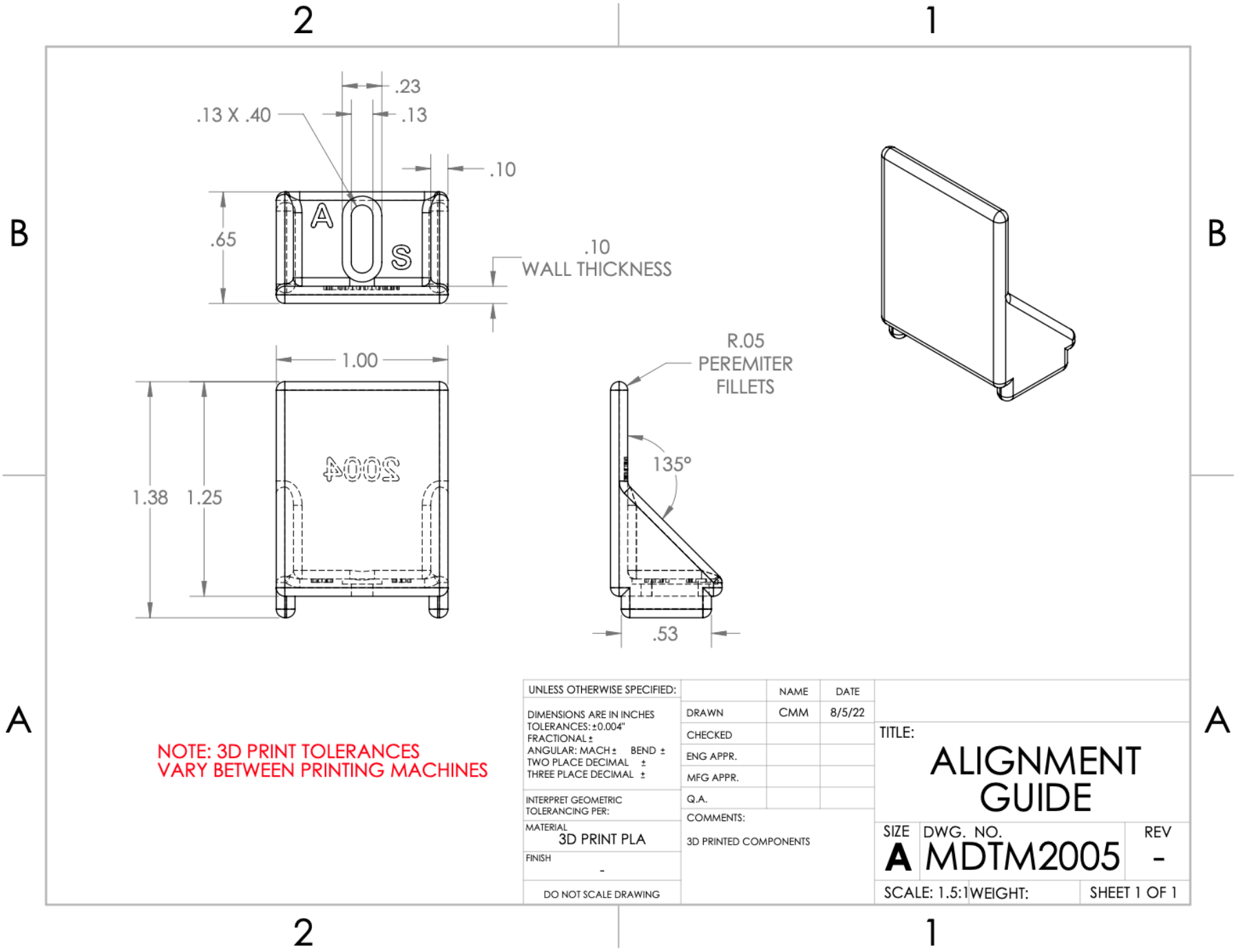

2
1
B
A
.250
0
.090
.188
SECTION A-A
R.063
A
A
∅.137 THRU
6X ∅ .157 ↧ .170
4X ∅ .134 THRU ALL
⌴ ∅ .264 ↧ .065
R.125
INTERIOR POCKET RADIUSES
NO SMALLER THAN 0.125"
4X ∅ .157 THRU ALL
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENTS
TITLE:
SMALL ENCLOSURE COVER
SIZE
A
DWG. NO.
MDTM2002
REV
-
SCALE: 1:1.5
WEIGHT:
SHEET 1 OF 1


2
1
B
A
.200
∅.060↧.049
0.1" STD. PIN SPACING
.963
.496
.338
.180
0
Arduino L.
.317
.069
WALL THICKNESS
1.717
.090
2.754
.216
1.076
1.149
1.377
1.602
2.641
0
.113
.249
.428
.393
.728
.951
2002
3 x ∅.126 THRU
.113
PEREMITER FILLETS
NOTE: 3D PRINT
TOLERANCES VARY
BETWEEN MACHINES
ARDUINO CONFIGURATION
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LEFT SIDE COVER
SIZE
A
DWG. NO.
MDTM2003
REV
-
SCALE: 1:1
WEIGHT:
SHEET 1 OF 2

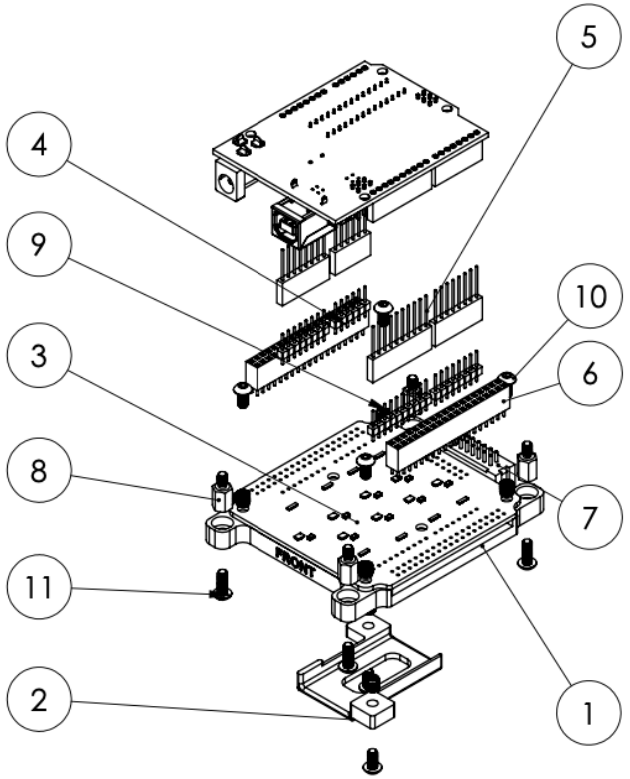

2
1
B
A
R.113
PEREMITER FILLETS
.951
.272
0
STM 1.
3 x ∅.126 THRU
2.057
.090
.468
.357
.049
WALL THICKNESS
.220
HEADER CLEARENCE
1.076
0
.125
.482
.729
1.149
1.377
1.602
2.268
2.441
2.641
0
.113
.269
.439
.459
.718
.963
2002
STM CONFIGURATION
NOTE: 3D PRINT TOLERANCES VARY BETWEEN MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
LEFT SIDE
HEADER COVER
SIZE
A
DWG. NO.
MDTM2003
REV
-
SCALE: 1:1
WEIGHT:
SHEET 2 OF 2

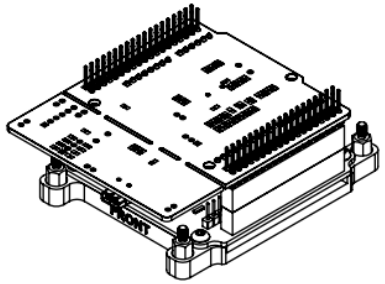

R.113
Peremiter Fillets
3 x ∅.126 Thru
18x ∅.060
STD. Pin Spacing
.951
.338
.272
.113
0
.160
.453
Arduino R.
.216
.090
0
.397
2.473
2.754
.291
1.076
.113
.303
.466
1.152
1.605
2.281
2.404
2.629
2.641
.252
.425
.449
.728
2003
RIGHT COVER IN ARDUINO CONFIGURATON
TOLERANCES VARY BETWEEN 3D PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
RIGHT HEADER COVER
SIZE
A
DWG. NO.
MDTM2004
REV
-
SCALE: 1:1
WEIGHT:
SHEET 1 OF 2


R.113
PEREMITER FILLETS
.951
.787
.390
0
STM R.
2 X 19 ∅.060
STD. PINOUT SPACING
0
.263
2.321
2.754
.468
.090
0
.357
1.076
0
.113
.312
.486
1.152
1.377
1.605
2.271
2.385
2.641
0
.113
.269
.459
.459
.718
.951
.963
2003
3X ∅ .126 ↧ .160
STM R.
TOLERANCES VARY BETWEEN 3D PRINTING MACHINES
RIGHT COVER IN
STM CONFIGURATION
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
RIGHT HEADER
COVER
SIZE
A
DWG. NO.
MDTM2004
REV
-
SCALE: 1:1
WEIGHT:
SHEET 2 OF 2


2
1
B
A
.23
.13 X .40
.13
.10
.65
.10
WALL THICKNESS
1.00
1.38
1.25
R.05
PEREMITER
FILLETS
135°
.53
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
3D PRINT PLA
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENTS
TITLE:
ALIGNMENT GUIDE
SIZE
A
DWG. NO.
MDTM2005
REV
-
SCALE: 1.5:1
WEIGHT:
SHEET 1 OF 1

ITEM NO.	PART NUMBER	DESCRIPTION	Material	Vendor	QTY.
1	MDTM3001	PCB Mounting Plate	ABS	In House	1
2	MDTM3002	RIBBON CABLE RETENTION BRACKET	PLA	IN HOUSE	1
3	Small PCB	Little Foot PCB	Printed Circuit Board	PCBway	1
4	PRPC040SACN-RC	40 x 1 BREAK AWAY MALE HEADER	-	DigiKey	32
5	0085	Arduino Header Extension	-	Adafruit	1
6	2222	2x19_3mm Female Header	-	ADAFRUIT	2
7	DS1013-16S	2X8 RIBBON CABLE PLUG-IN	-	DIGIKEY	1
8	93655A007	M3x0.5_4.7mm Standoff	Stainless Steel	McMaster Carr	4
9	4255	Heat-Set Inserts for Plastic	Brass	Adafruit	6
10	92095A179	M3 x 0.5 6mm BHS	18-8 Stainless Steel	McMaster-Carr	6
11	92095A181	M3 x 0.5 8mm BHS	18-8 Stainless Steel	McMaster-Carr	4





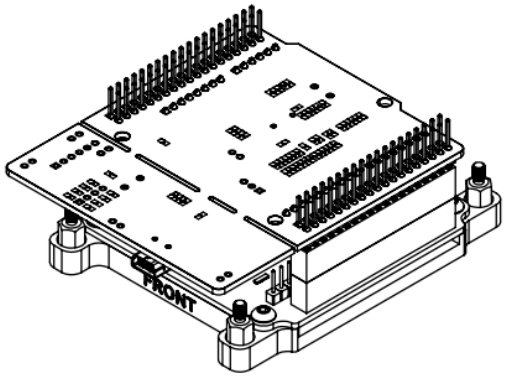
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL -	COMMENTS:		
FINISH			
DO NOT SCALE DRAWING			

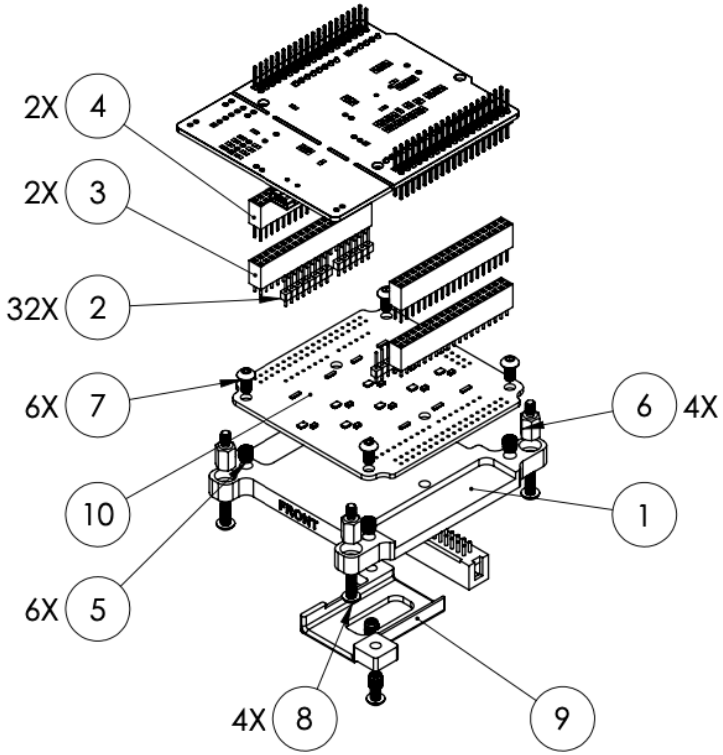
TITLE:

ARD. TOOL PACK CONFIGURATION

SIZE	DWG. NO.	REV
A	MDTA3000	
SCALE: 1:2	WEIGHT:	SHEET 2 OF 2

ITEM NO.	PART NUMBER	DESCRIPTION	MATERIAL	VENDOR	QTY.
1	MDTM3001	PCB MOUNTING PLATE	ABS	IN HOUSE	1
2	PRPC040SACN-RC	40 X 1 BREAK AWAY MALE HEADER	-	DIGIKEY	32
3	2222	2X19_3mm FEMALE HEADER	-	ADAFRUIT	2
4	4079	2X19_6mm FEMALE HEADER	-	ADAFRUIT	2
5	4255	HEAT-SET INSERTS FOR PLASTIC	BRASS	ADAFRUIT	6
6	93655A007	M3x0.5_4.7mm Standoff	Stainless Steel	McMaster Carr	4
7	92095A179	M3 X 0.5 6mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	6
8	92095A181	M3 X 0.5 8mm BHS	18-8 STAINLESS STEEL	MCMASTER-CARR	4
9	MDTM3002	RIBBON CABLE RETENTION BRACKET	PLA	IN HOUSE	1
10	SMALL PCB	LITTLE FOOT PCB	PRINTED CIRCUIT BOARD	PCBWAY	1



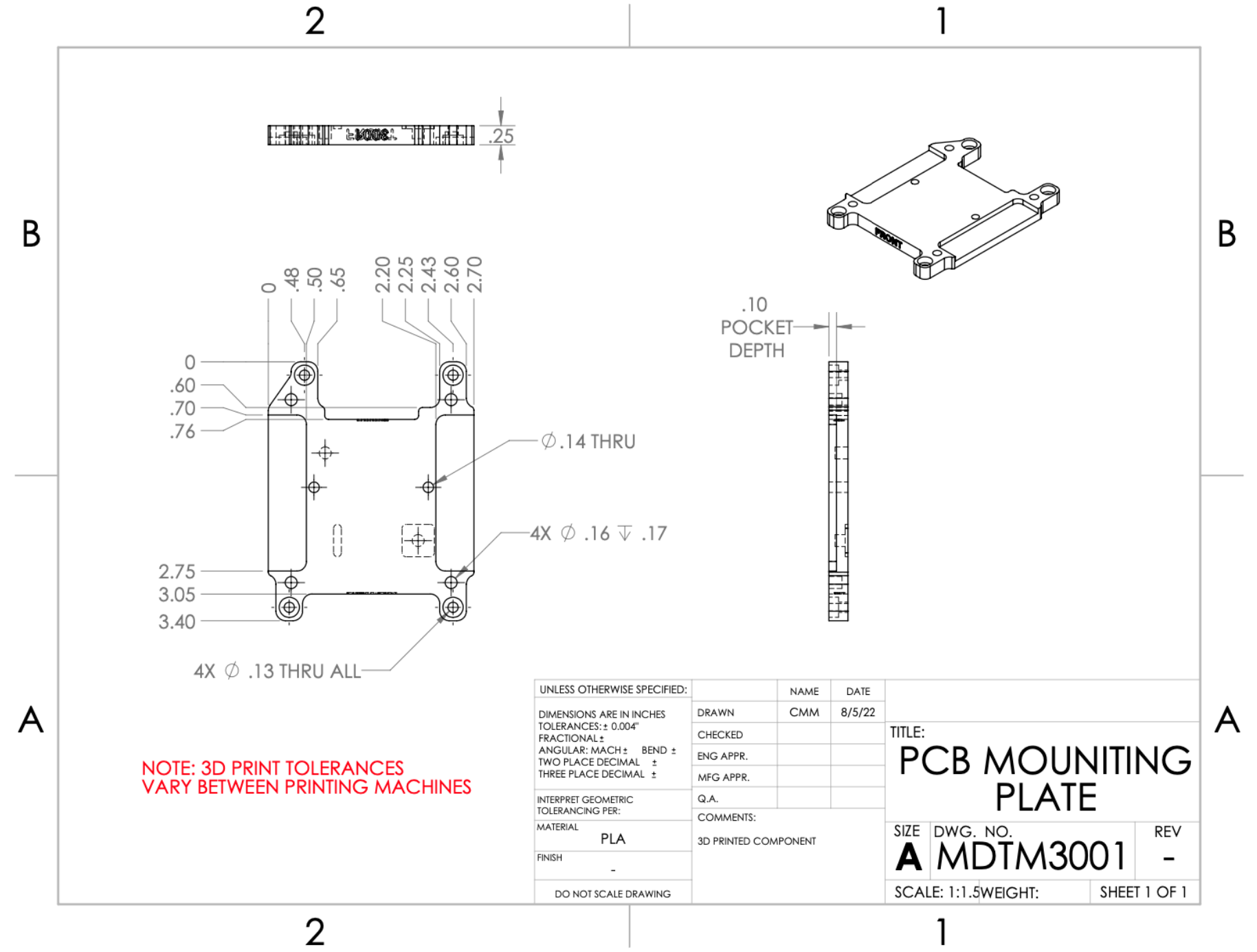


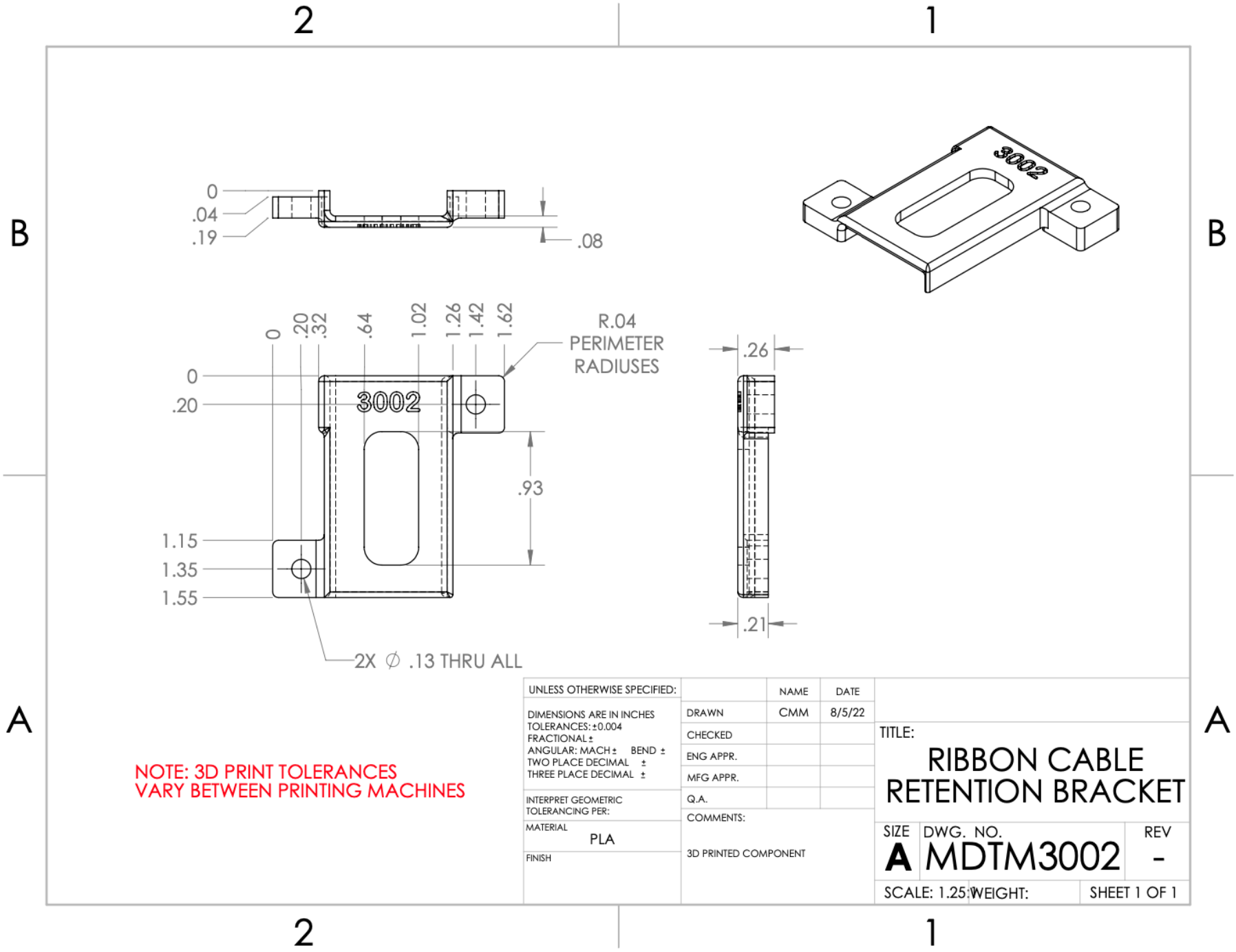
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL -	COMMENTS:		
FINISH			
DO NOT SCALE DRAWING			

TITLE:

STM TOOL PACK CONFIGURATION

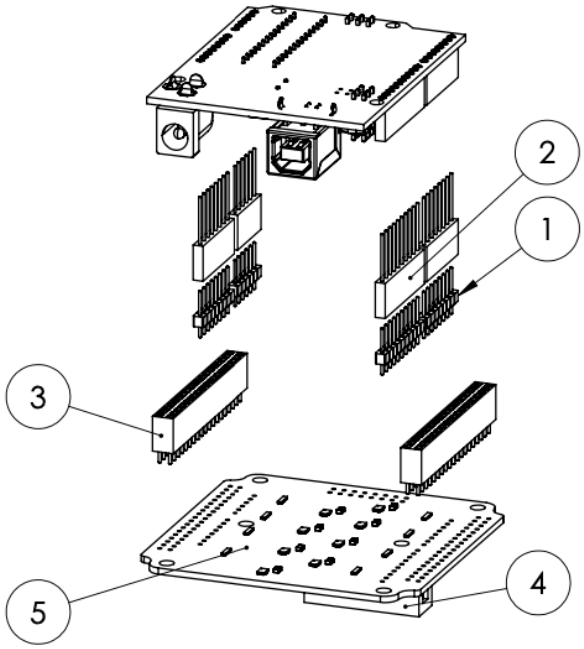
SIZE	DWG. NO.	REV
A	MDTA3000	-
SCALE: 1:1.5	WEIGHT:	SHEET 1 OF 2

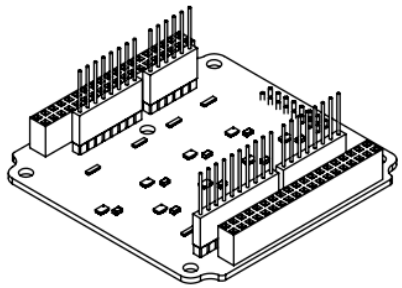

2
1
B
A
.25
0
.48
.50
.65
2.20
2.25
2.43
2.60
2.70
0
.60
.70
.76
2.75
3.05
3.40
∅.14 THRU
4X ∅ .16 ↧ .17
4X ∅ .13 THRU ALL
.10
POCKET
DEPTH
FRONT
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ± 0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
PCB MOUNITING PLATE
SIZE
A
DWG. NO.
MDTM3001
REV
-
SCALE: 1:1.5
WEIGHT:
SHEET 1 OF 1


2
1
B
A
0
.04
.19
.08
3002
0
.20
.32
.64
1.02
1.26
1.42
1.62
R.04
PERIMETER
RADIUSES
.26
.93
1.15
1.35
1.55
.21
2X ∅ .13 THRU ALL
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
RIBBON CABLE
RETENTION BRACKET
SIZE
A
DWG. NO.
MDTM3002
REV
-
SCALE: 1.25
WEIGHT:
SHEET 1 OF 1

ITEM NO.	PART NUMBER	DESCRIPTION	Material	Vendor	QTY.
1	PRPC040SACN-RC	40 x 1 BREAK AWAY MALE HEADER	-	DigiKey	32
2	0085	Arduino Header Extension	-	Adafruit	1
3	2222	2x19_3mm Female Header	-	ADAFRUIT	2
4	DS1013-16S	2X8 RIBBON CABLE PLUG-IN	-	DIGIKEY	1
5	Small PCB	Little Foot PCB	Printed Circuit Board	PCBway	1

ASSEMBLY INSTRUCTIONS:
BREAK MALE HEADER PINS INTO:
(2) 1X8, (1) 1X6, (1) 1X10 SECTIONS
SOLDER EACH SECTION TO RESPECTIVE PIN HOLES ON PCB
SOLDER 2X19 HEADERS TO PCB
SOLDER RIBBON CABLE PLUG TO UNDERSIDE OF PCB
INSERT REMOVABLE HEADERS OVER MALE HEADERS





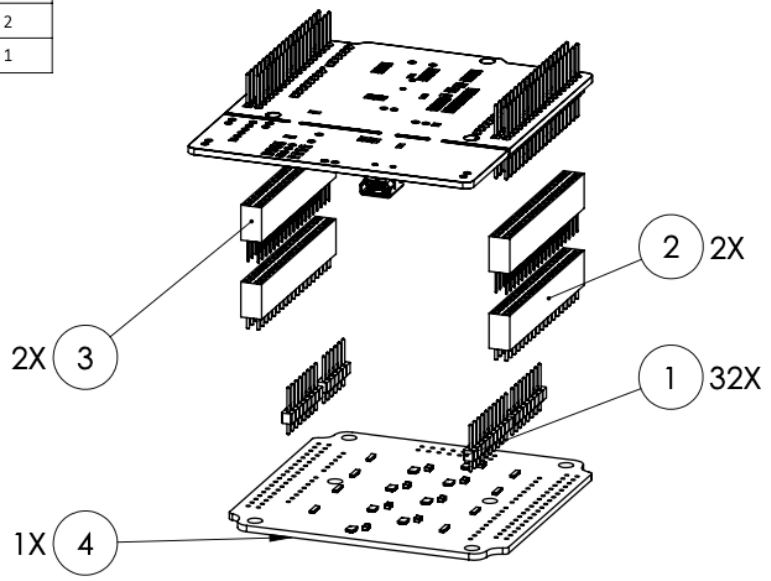
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL N/A	ARDUINO CONFIGURATION SHOWN		
FINISH N/A			

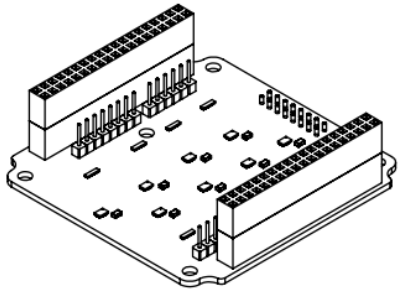
TITLE: ARDUINO INTERFACE

SIZE	DWG. NO.	REV
A	MDTA3100	-
SCALE: 1:1.5	WEIGHT:	SHEET 1 OF 2

ITEM NO.	PART NUMBER	DESCRIPTION	Material	Vendor	QTY.
1	PRPC040SACN-RC	40 x 1 BREAK AWAY MALE HEADER	-	DigiKey	32
2	2222	2x19_3mm Female Header	-	ADAFRUIT	2
3	4079	2x19_6mm Female Header	-	Adafruit	2
4	Small PCB	Little Foot PCB	Printed Circuit Board	PCBway	1

ASSEMBLY INSTRUCTIONS:
BREAK MALE HEADER PINS INTO:
(2) 1X8, (1) 1X6, (1) 1X10 SECTIONS
SOLDER EACH SECTION TO RESPECTIVE PIN HOLES ON PCB
SOLDER 2X19 HEADERS TO PCB
SOLDER RIBBON CABLE PLUG TO UNDERSIDE OF PCB
INSERT REMOVABLE HEADERS OVER MALE HEADERS



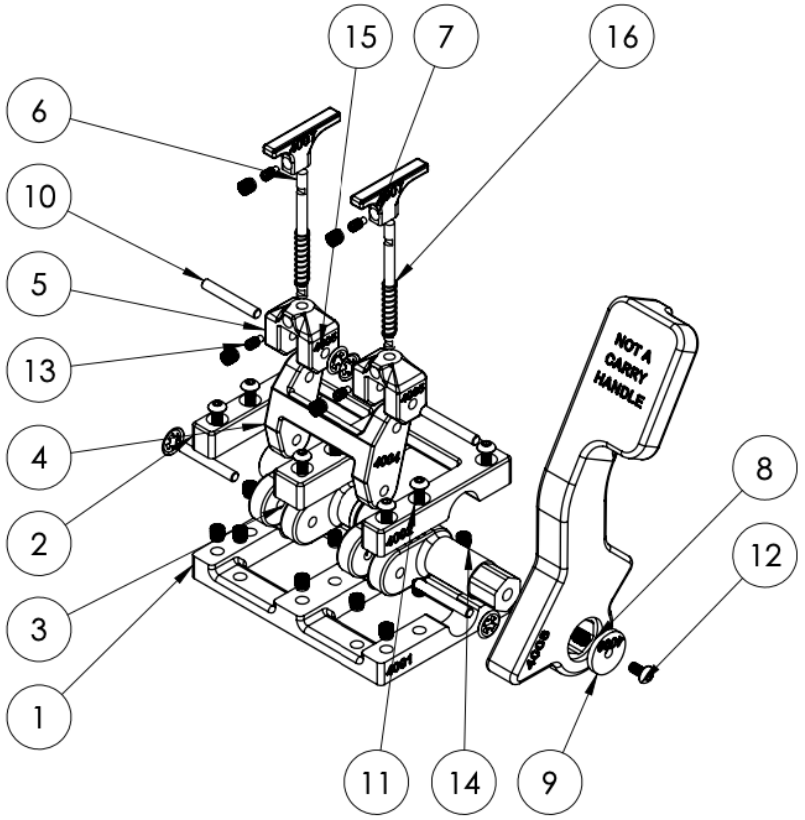


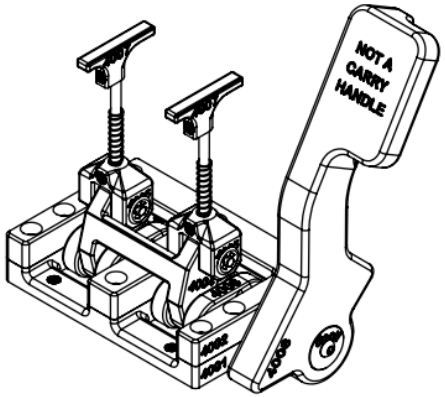
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
	Q.A.		
INTERPRET GEOMETRIC TOLERANCING PER:	COMMENTS:		
MATERIAL -	STM CONFIGURATION SHOWN		
FINISH -			

TITLE:
SMALL PCB INTERFACE

SIZE	DWG. NO.	REV
A	MDTA3100	-
SCALE: 1:1.5	WEIGHT:	SHEET 2 OF 2

ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	MDTM4001	Lower Race	1
2	MDTM4002	Upper Race	1
3	MDTM4003	Crank	1
4	MDTM4004	Merged Link	1
5	MDTM4005	Push Rod Foot	2
6	89535K16	1/8" Steel Rod 1.83" Length	2
7	MDTM4007	Push Rod Hat	2
8	MDTM4008	Crank Handle	1
9	MDTM4009	Crank Retaining Washer	1
10	89535K16	1/8" Steel Rod 0.9" Length	4
11	92095A181	M3 x 0.5 8mm BHS	9
12	92095A179	M3 x 0.5 6mm BHS	1
13	92905A602	Alloy Steel Extended-Tip Set Screw	4
14	4255	Heat-Set Inserts for Plastic	18
15	98430A116	Push-on External Retaining Rings	8
16	9434K23	Music-Wire Steel Compression Springs	2



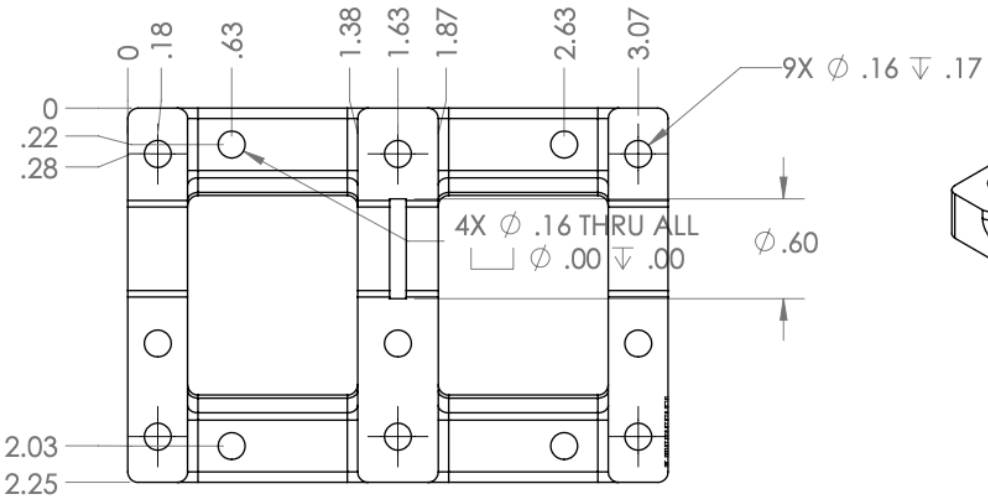


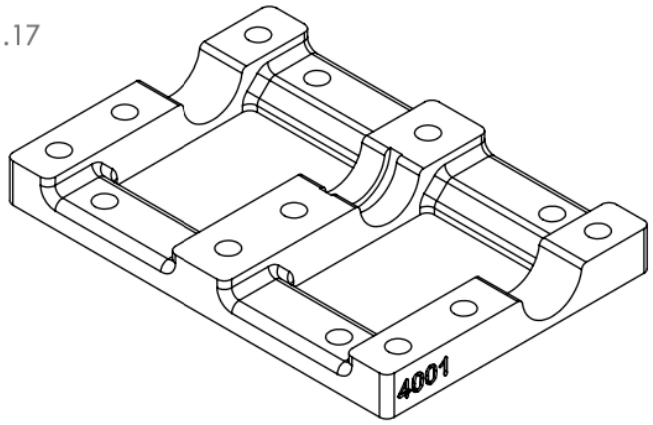
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL N/A	COMMENTS: NESTED IN MDTA2000		
FINISH N/A			
DO NOT SCALE DRAWING			

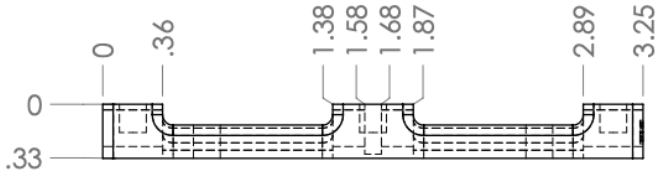
TITLE:

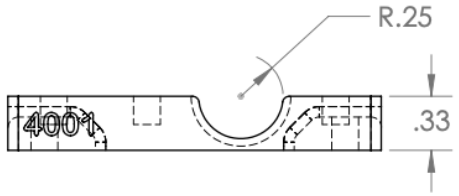
EJECTION CRANK ASM

SIZE	DWG. NO.	REV
A	MDTA4000	-
SCALE: 1:2	WEIGHT:	SHEET 1 OF 1







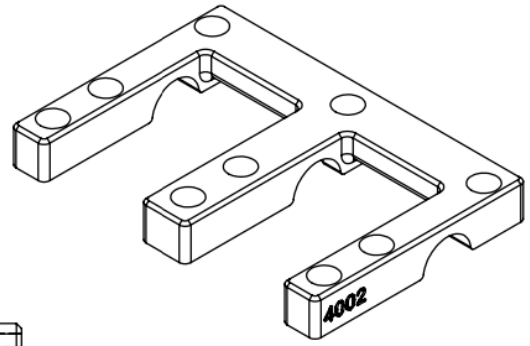


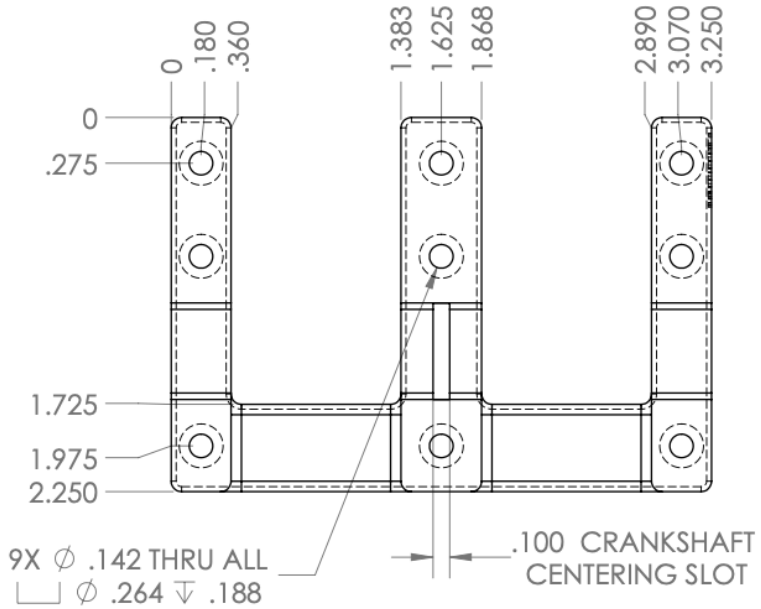
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINT MACHINES

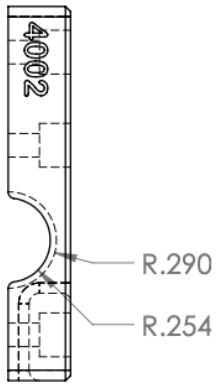
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL CARBON FIBER NYLON WITH CONTINUOUS FIBERGLASS STRAND	COMMENTS: 3D PRINTED COMPONENT		

TITLE:
LOWER CRANK HOUSING

SIZE	DWG. NO.	REV
A	MDTM4001	-
SCALE: 1:1	WEIGHT:	SHEET 1 OF 1





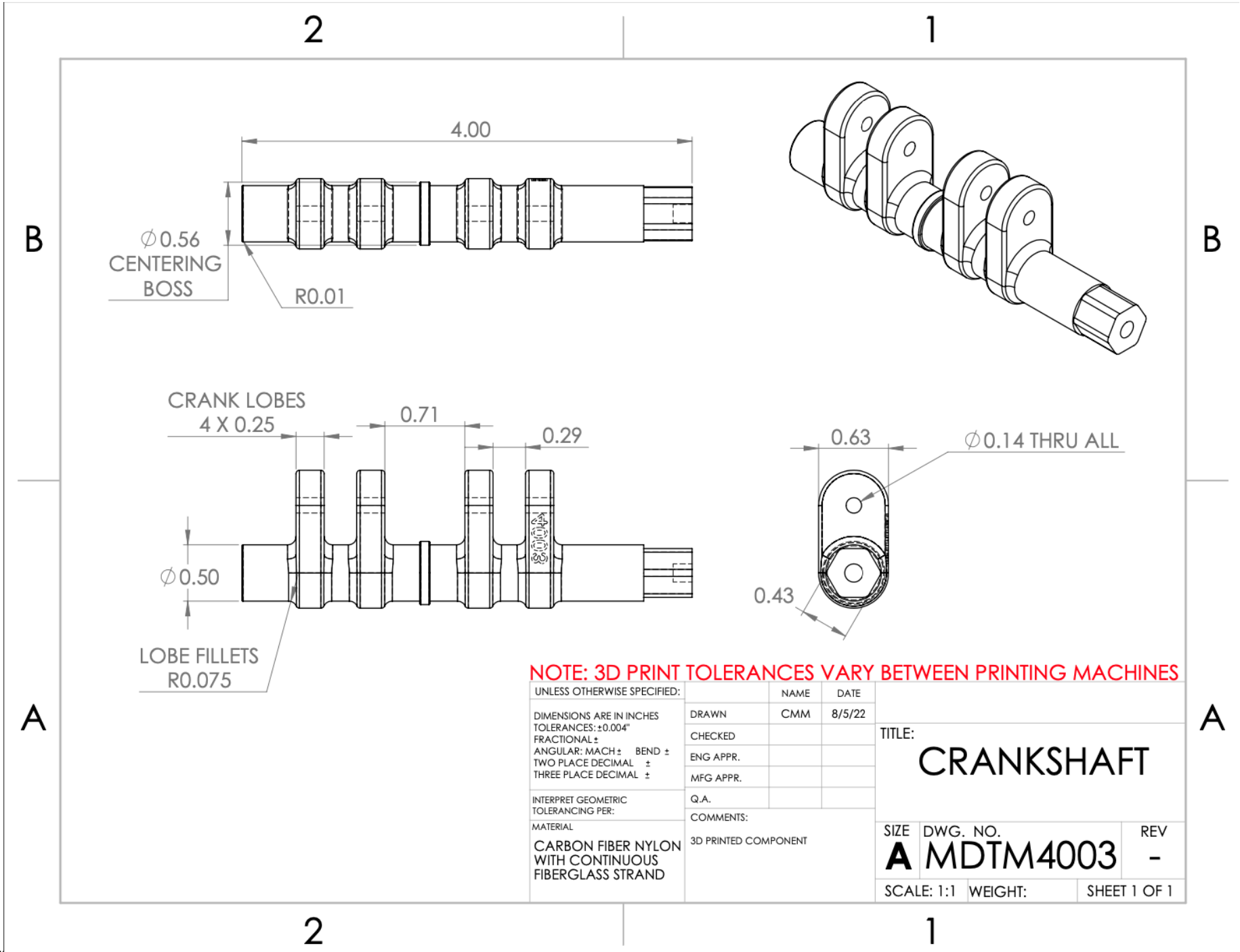


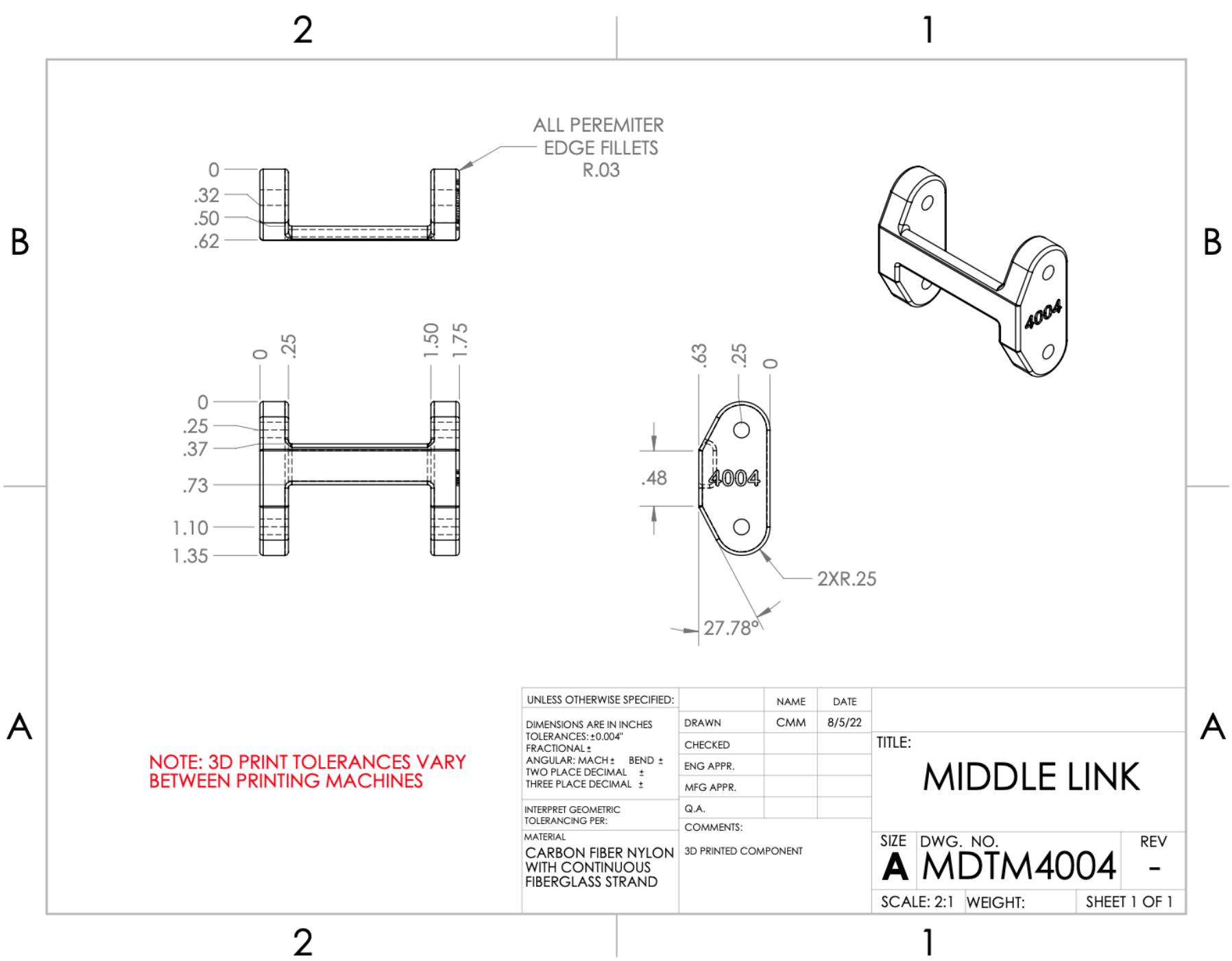
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES

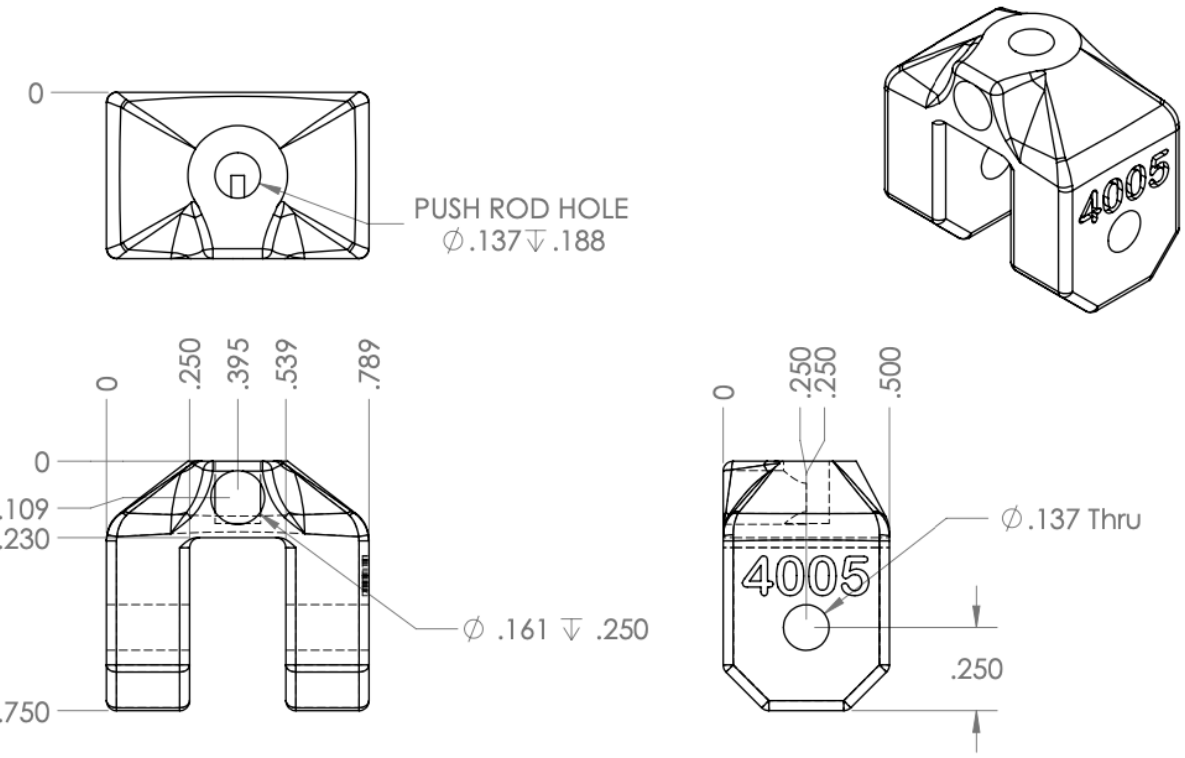
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL CARBON FIBER NYLON WITH CONTINUOUS FIBER STRAND	COMMENTS: 3D PRINTED COMPONENT		

TITLE:
UPPER CRANK HOUSING

SIZE	DWG. NO.	REV
A	MDTM4002	-
SCALE: 1:1	WEIGHT:	SHEET 1 OF 1


2
1
B
A
4.00
∅0.56
CENTERING
BOSS
R0.01
CRANK LOBES
4 X 0.25
0.71
0.29
∅0.50
LOBE FILLETS
R0.075
0.63
∅0.14 THRU ALL
0.43
NOTE: 3D PRINT TOLERANCES VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON WITH CONTINUOUS FIBERGLASS STRAND
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
CRANKSHAFT
SIZE
A
DWG. NO.
MDTM4003
REV
-
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1


2
1
B
A
ALL PEREMITER
EDGE FILLETS
R.03
0
.32
.50
.62
0
.25
1.50
1.75
0
.25
.37
.73
1.10
1.35
.63
.25
0
.48
4004
2XR.25
27.78°
4004
NOTE: 3D PRINT TOLERANCES VARY
BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC
TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
WITH CONTINUOUS
FIBERGLASS STRAND
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
MIDDLE LINK
SIZE
A
DWG. NO.
MDTM4004
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1

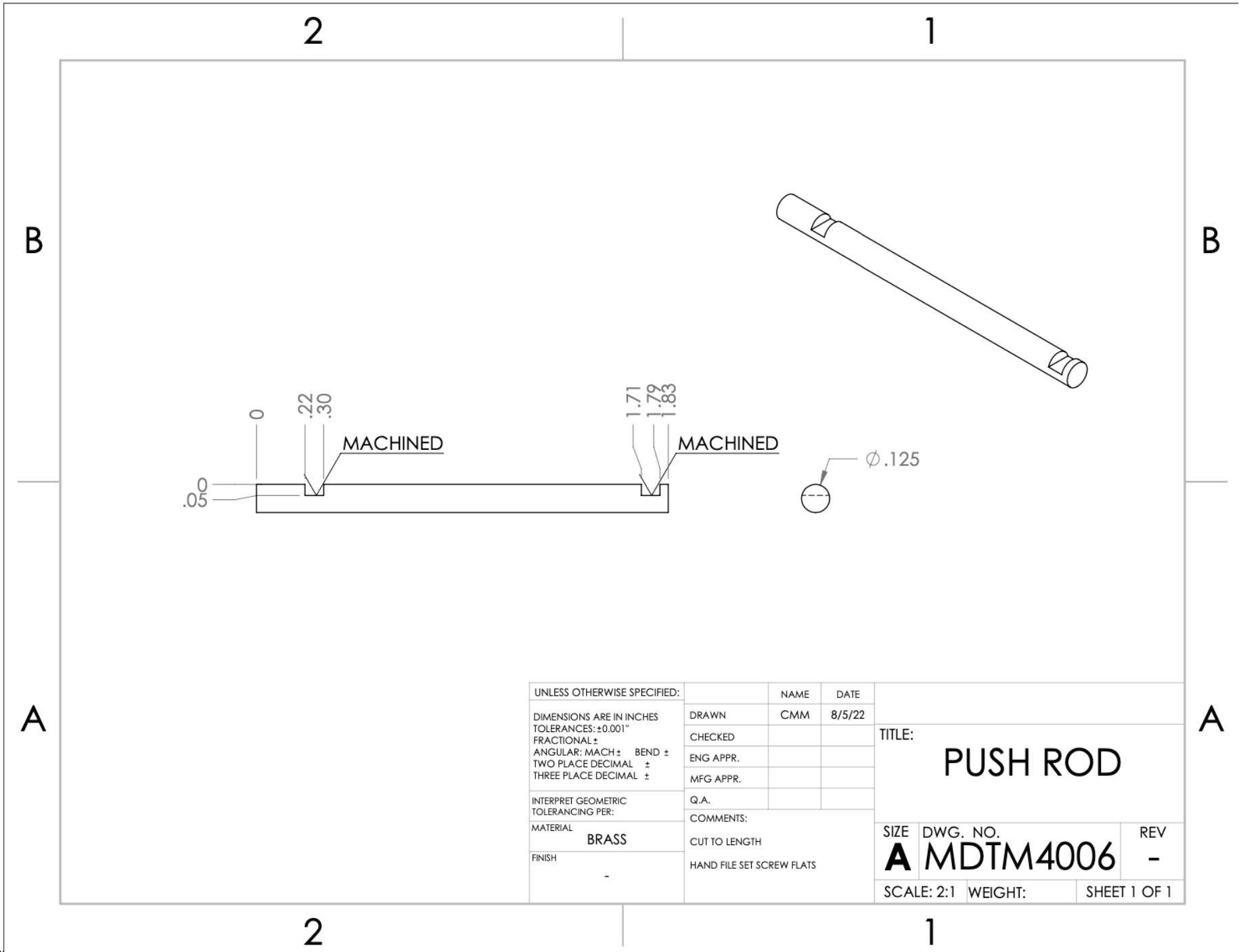


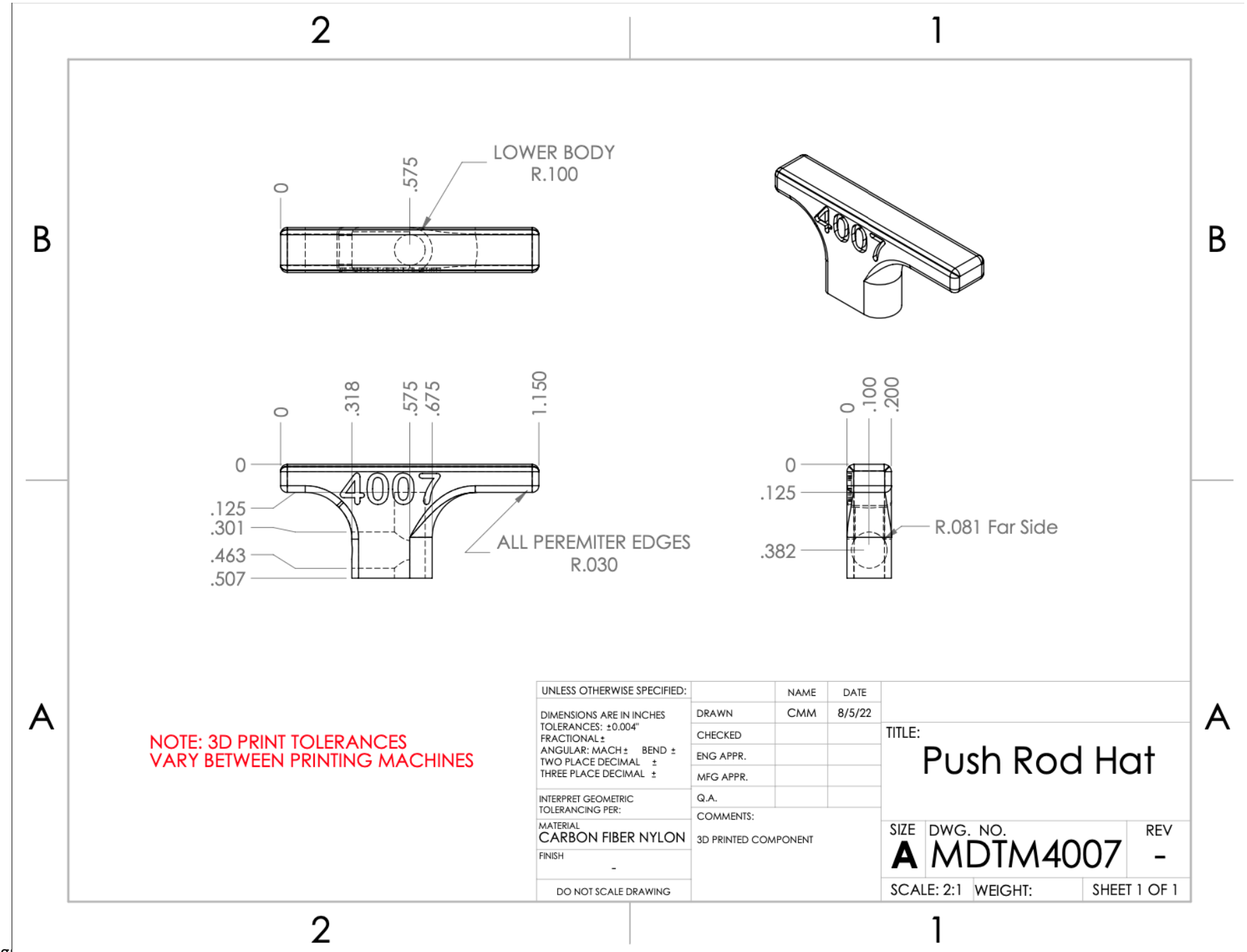
NOTE: 3D PRINT TOLERANCES VARY BETWEEN MACHINES

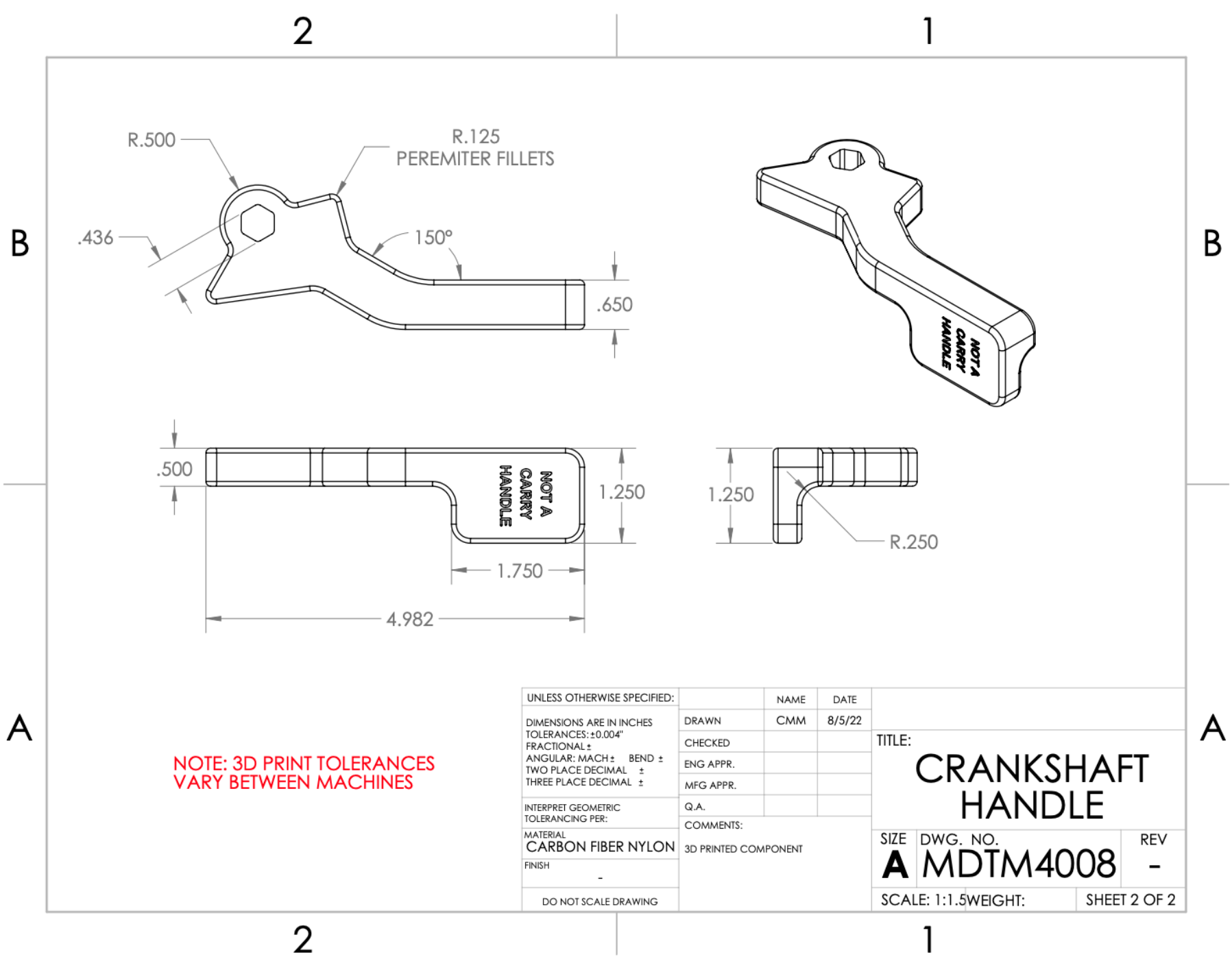
UNLESS OTHERWISE SPECIFIED:		NAME	DATE
DIMENSIONS ARE IN INCHES TOLERANCES: ±0.004" FRACTIONAL ± ANGULAR: MACH ± BEND ± TWO PLACE DECIMAL ± THREE PLACE DECIMAL ±	DRAWN	CMM	8/5/22
	CHECKED		
	ENG APPR.		
	MFG APPR.		
INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
MATERIAL CARBON FIBER NYLON	COMMENTS: 3D PRINTED COMPONENT		
FINISH -			
DO NOT SCALE DRAWING			

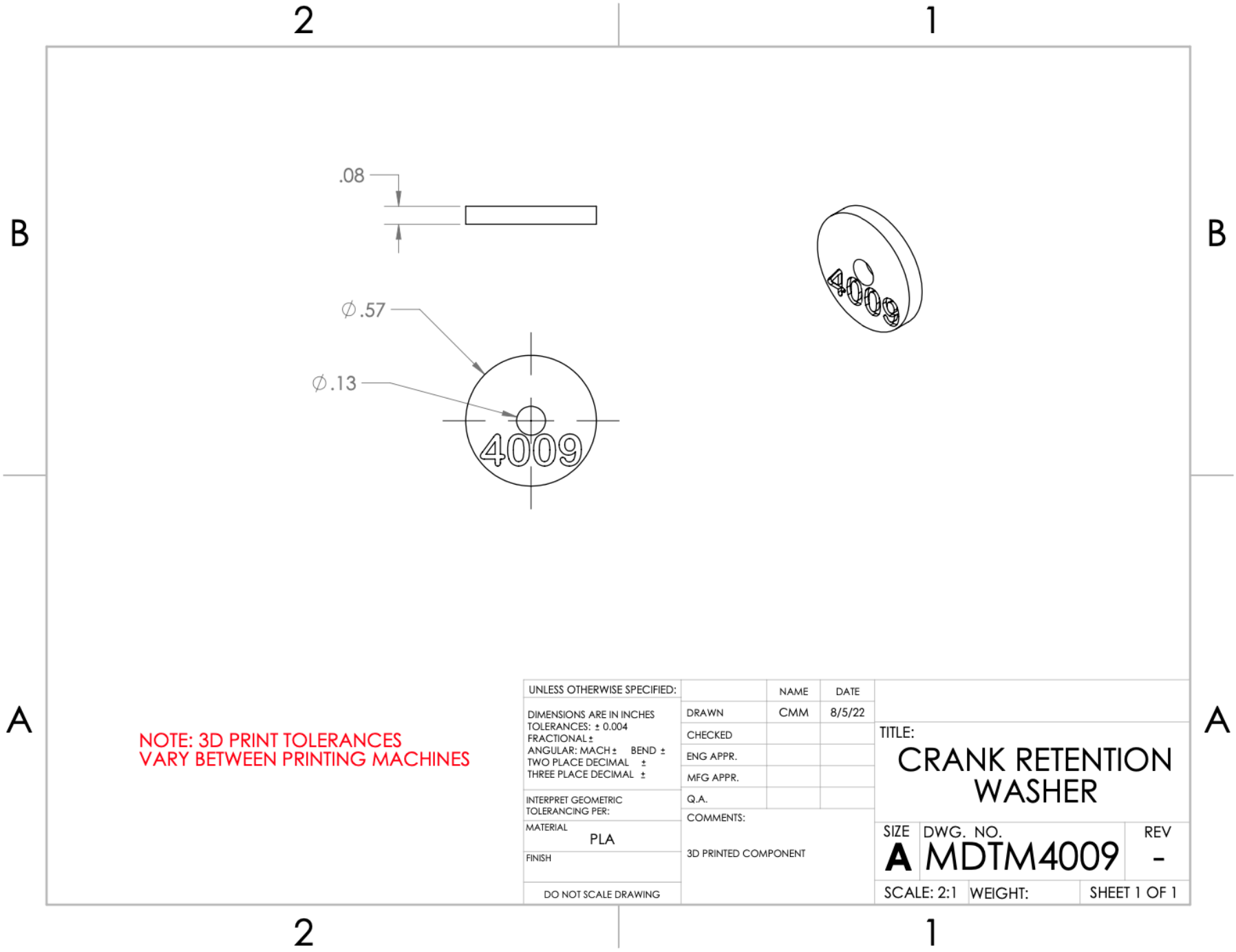
TITLE:
PUSH ROD RETENSION LINK

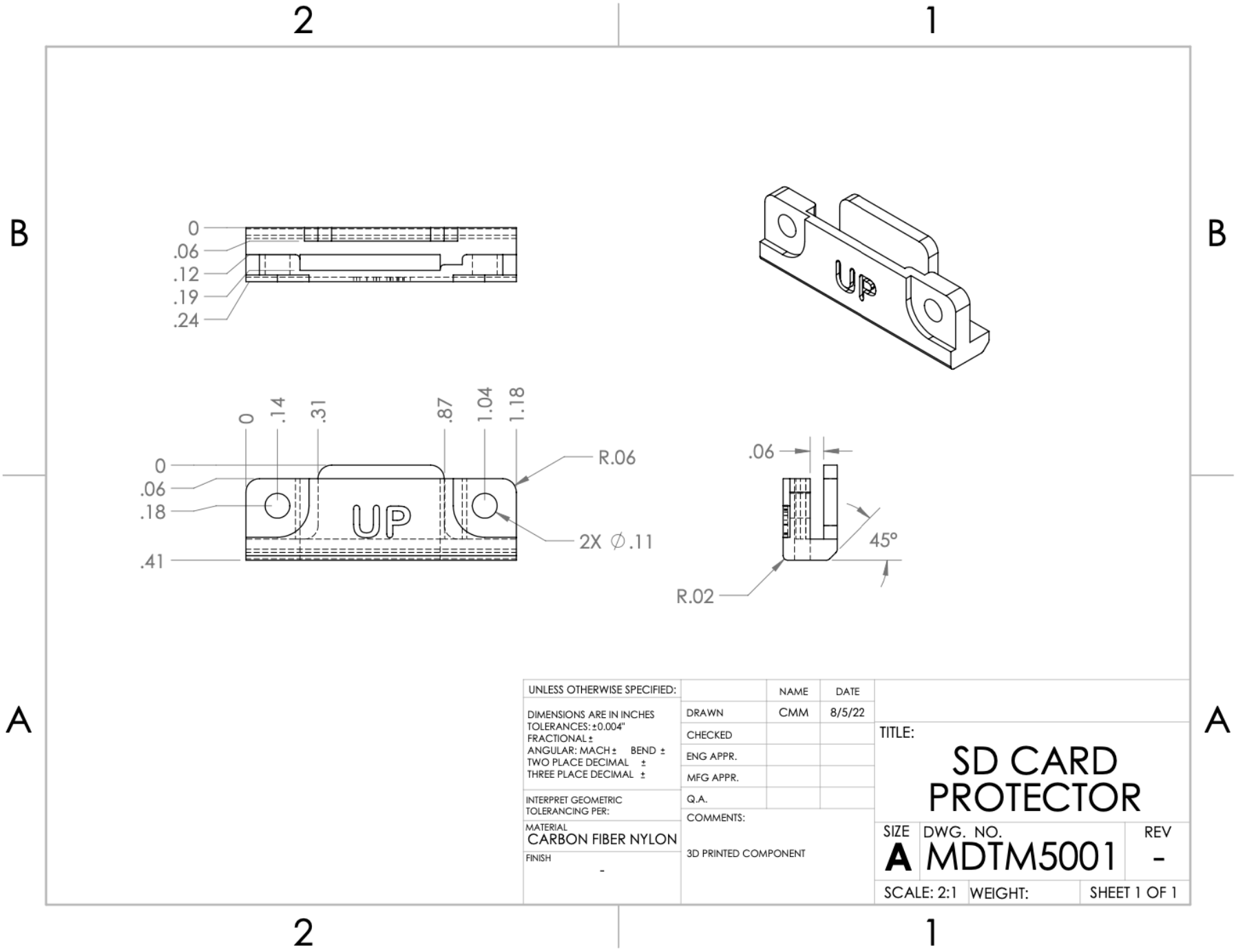
SIZE	DWG. NO.	REV
A	MDTM4005	-
SCALE:2:1	WEIGHT:	SHEET 1 OF 1


2
1
B
A
0
.22
.30
MACHINED
1.71
1.79
1.83
MACHINED
0
.05
∅.125
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.001"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
BRASS
FINISH
-
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
CUT TO LENGTH
HAND FILE SET SCREW FLATS
TITLE:
PUSH ROD
SIZE
A
DWG. NO.
MDTM4006
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1


2
1
B
A
LOWER BODY
R.100
0
.575
4007
0
.318
.575
.675
1.150
0
.125
.301
.463
.507
4007
ALL PEREMITER EDGES
R.030
0
.100
.200
0
.125
.382
R.081 Far Side
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
Push Rod Hat
SIZE
A
DWG. NO.
MDTM4007
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1


2
1
B
A
R.500
R.125
PEREMITER FILLETS
.436
150°
.650
NOT A CARRY HANDLE
.500
1.250
1.750
4.982
1.250
R.250
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
CRANKSHAFT HANDLE
SIZE
A
DWG. NO.
MDTM4008
REV
-
SCALE: 1:1.5
WEIGHT:
SHEET 2 OF 2


2
1
B
A
.08
∅.57
∅.13
4009
4009
NOTE: 3D PRINT TOLERANCES
VARY BETWEEN PRINTING MACHINES
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ± 0.004
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
PLA
FINISH
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
CRANK RETENTION WASHER
SIZE
A
DWG. NO.
MDTM4009
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1

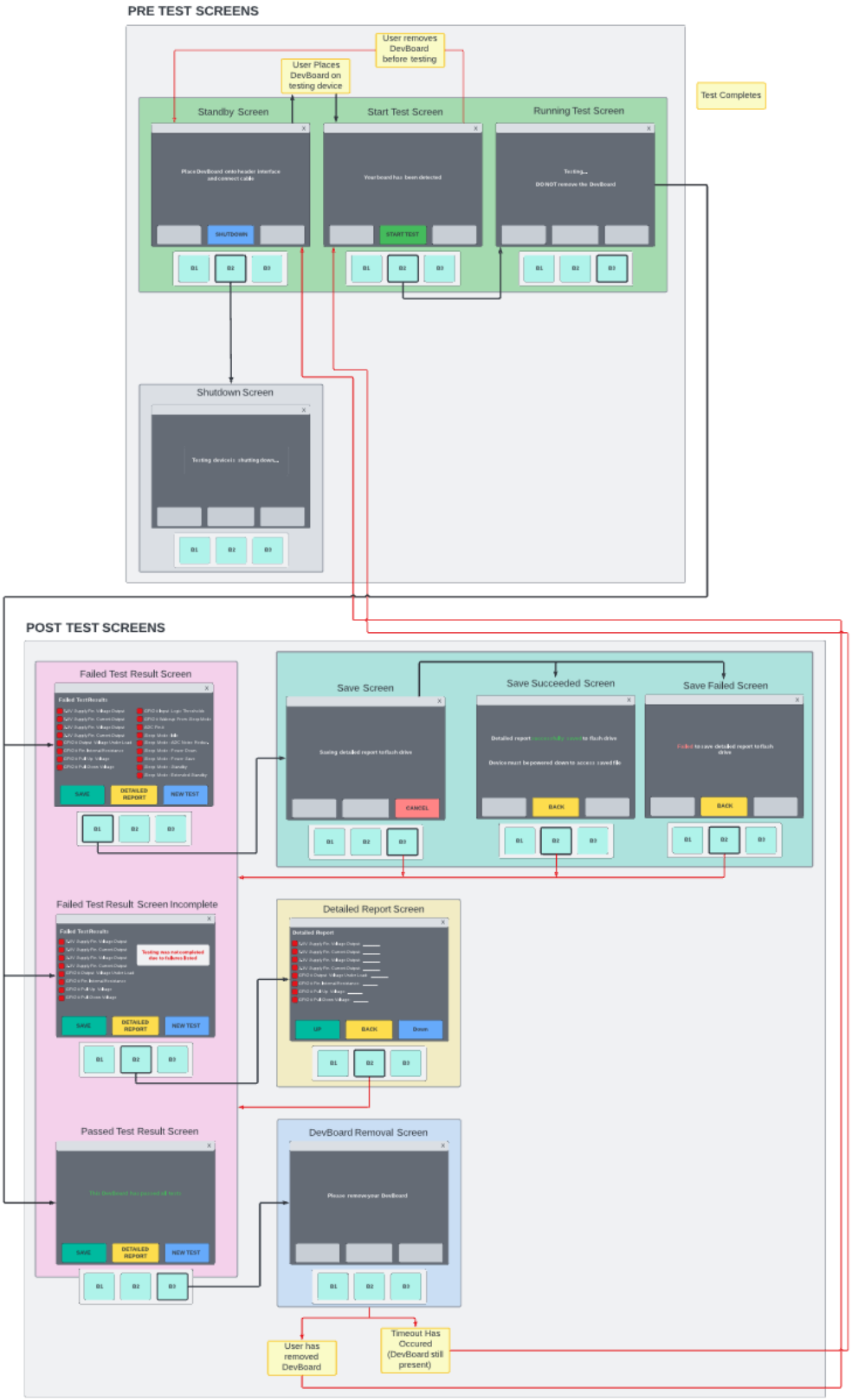

2
1
B
A
0
.06
.12
.19
.24
UP
0
.14
.31
.87
1.04
1.18
0
.06
.18
.41
R.06
2X ∅.11
UP
.06
45°
R.02
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES: ±0.004"
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC TOLERANCING PER:
MATERIAL
CARBON FIBER NYLON
FINISH
-
NAME
DATE
DRAWN
CMM
8/5/22
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
3D PRINTED COMPONENT
TITLE:
SD CARD PROTECTOR
SIZE
A
DWG. NO.
MDTM5001
REV
-
SCALE: 2:1
WEIGHT:
SHEET 1 OF 1

Appendix C - Requirements and Specifications

Physical Requirements			
REQ #	**REQUIREMENT**	**SPEC #**	**SPECIFICATION**
1.00	The device **must fit on a desktop**	1.00.1	The device must have overall dimensions not exceeding 1' x 1' x 1'
1.01	The device must be **portable**	1.01.1	The device must weigh less than 10 lbs
		1.01.2	The device must not need additional calibration when moved
1.02	The device must be physically **assembled into a single piece**	1.02.1	The device must not have unattached or unsecured components
1.03	The device must be **simple to load and unload without causing damage to the subject board**	1.03.1	The device must use two alignment posts to aid in loading and prevent damage
		1.03.2	The device must use a mechanism to easily and safely remove the subject board without causing damage
1.04	The device **must have an enclosure that can withstand the rigors of daily use**	1.04.1	The device must have 3D printed parts with an infill of at least 30%
Hardware Requirements			
2.00	The device must utilize a physical **switch to control main power**	2.00.1	The device must use a toggle switch to control the main power supply
2.01	The device must be powered from a **120V AC wall outlet**	2.01.1	The device must use a 120V AC
2.02	The device must implement **circuit protection to avoid damage** to its hardware	2.02.1	The device must utilize overvoltage / surge protection IC's for circuits that have potential to experience overvoltage
2.03	The device must **only require the subject board** from the operator for conduct testing	N/A	N/A
2.04	The device must have a **servicable hardware interface** to the subject board	2.04.1	Must have a header that connects to the I/O pins of the subject board
2.05	The device must **accurately measure** the subject board **voltages** under test	2.05.1	The device must implement hardware that measures voltage with an accuracy of 0.1V within the range 0 to 5.5 V unless otherwise specified
2.06	The device must **accurately measure** the subject board **currents** under test	2.06.1	The device must implement hardware that measures current with an accuracy of 10 uA within the range 0 to 1000uA (subject to change based on IC)
		2.06.2	The device must implement hardware that measures current with an accuracy of 10 mA within the range 0 to 1 A (subject to change based on IC)
Testing Requirements			
3.00	The device must test and record the **voltage supplied** from the subject board's **5V and 3.3V output pins**	3.00.1	The device must test that the supply voltage is within the configurable range
3.02	The device must test and record each GPIO pin's output voltage under **load sourcing**	3.02.1	The device must test that the pin's output voltage remains above a configurable threshold
3.03	The device must test and record each GPIO pin's internal resistor value	3.03.1	The device must test that the calculated internal resistance of each GPIO pin is within the manufacturers range
3.04	The device must test and record each GPIO pin's internally resistive **pull-up voltage while unloaded**	3.04.1	The device must measure voltage from each GPIO to GND while pin is configured as pull-up
		3.04.2	The device must measure and test each GPIO pin's voltage while configured as pull-up to be within a configurable range
3.05	The device must test and record each GPIO pin's internally resistive **pull-down voltage while unloaded**	3.05.1	The device must measure and test each GPIO pin's voltage while configured as pull-down to be within a configurable range
3.06	The device must test and record the GPIO **input pin logic thresholds** of the subject board	3.06.1	The device must test that logic low/high is produced when given a voltage within their threshold
3.07	The device must test and record the **accuracy** of the subject board's **ADC pins and channels**	3.07.1	The device must generate voltages for the subject board's ADC pins
		3.07.2	The device must test the measured voltages from the ADC pins
3.08	The device must test and record **all of the pins** capable of **GPIO** wakeups from **sleep mode**	3.08.1	The device must generate a signal to wake the subject board
		3.08.2	The device must test that the subject board's GPIO pins can exit sleep mode
3.09	The device must test all **sleep modes and power modes**	3.09.1	The device test the subject board's current draw during sleep and power modes to be below a configurable threshold
Software Requirements			
4.00	The device must **identify the subject board being tested**	4.00.1	The device's software must read and record the subject board's identification information found in the Device Descriptor Table
4.01	The GUI implemented on the device must be **intuitive to use**	4.01.1	Any interactive icons on the device's GUI must be labeled
4.02	The device must allow the user to **start** a **test through** its **GUI**	4.02.1	The device's GUI must display a "begin test" button once the subject board is properly loaded
4.03	The device's GUI must **provide a detailed report** of the subject board's test	4.03.1	The device's GUI must display a report of the failures of the subject board at the conclusion of testing
		4.03.2	The device must report if an error occurred in the execution of a test

4.04	The device's software must **allow configurable thresholds for tests**	4.04.1	The device's software must use removable media to load a configuration file
4.05	The software developed for the device must be **appropriately documented**	4.05.1	The device's software must be commented
		4.05.2	The device's software must be documented with a manual
STM32 Requirements			
5.00	The device **production quantity must be 5**	5.00.1	There must be 5 total STM testing devices produced
Arduino Requirements			
6.00	The device **production quantity must be 5**	6.00.1	There must be 5 total Arduino testing devices produced

Appendix D - Graphical User Interface



Appendix E - LTSpice Simulations

Current IC simulation for measuring 0-1A with 10mA resolution

In the following simulation, Raspberry Pi GPIO signals are represented by 3.3V voltage sources. A variable load resistor was used to simulate the Subject Board drawing variable amount of current. The simulation was executed and the voltage drop across the shunt resistor was plotted as seen below.

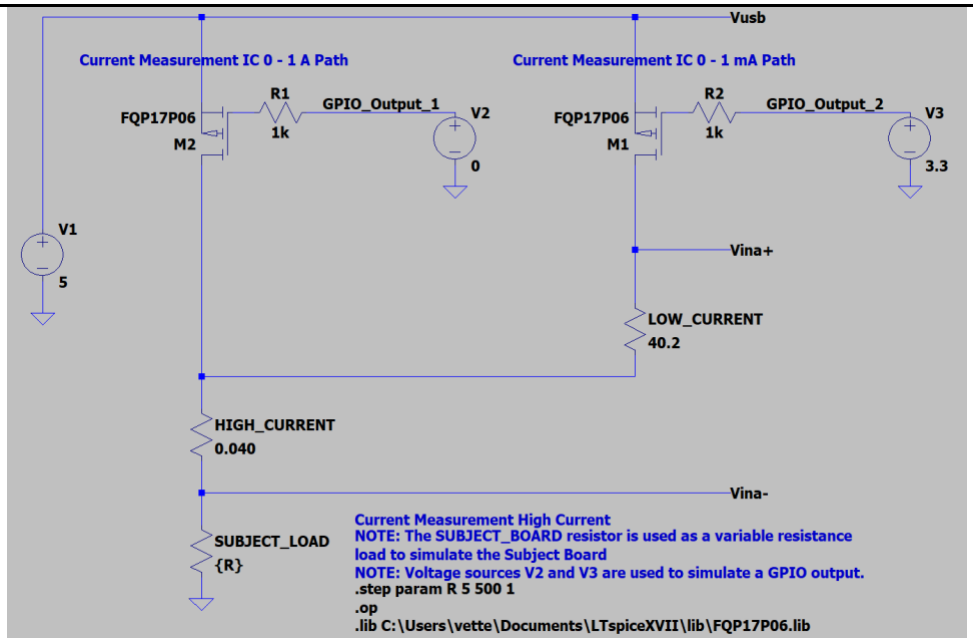


Figure E1: Current IC simulation circuit

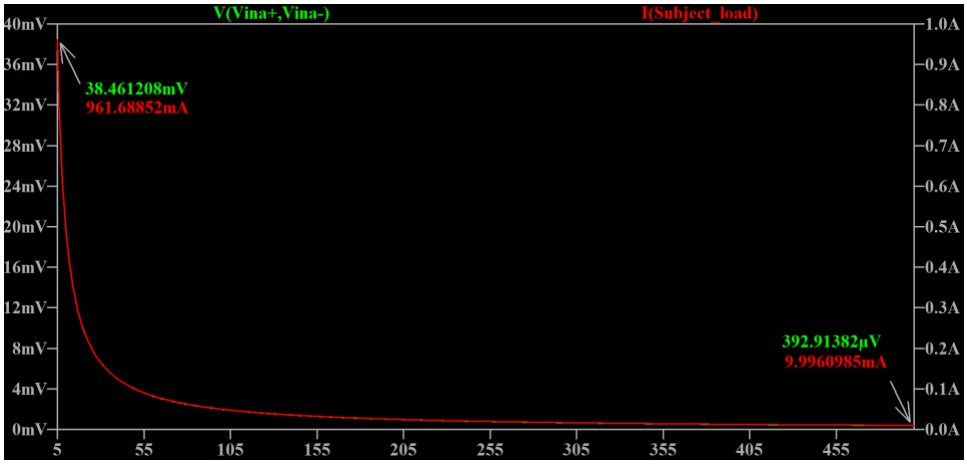


Figure E2: Current IC simulation

From these simulated results, the voltage drop seen by the INA219BID comes out to 38.46 mV at 961.6 mA load and 392.9 uV at 9.996 mA load, which aligns with calculations made within 5% error.

Current IC simulation for measuring 0-1mA with 10µA resolution

The hand calculations are also confirmed by the following LTSpice simulation:

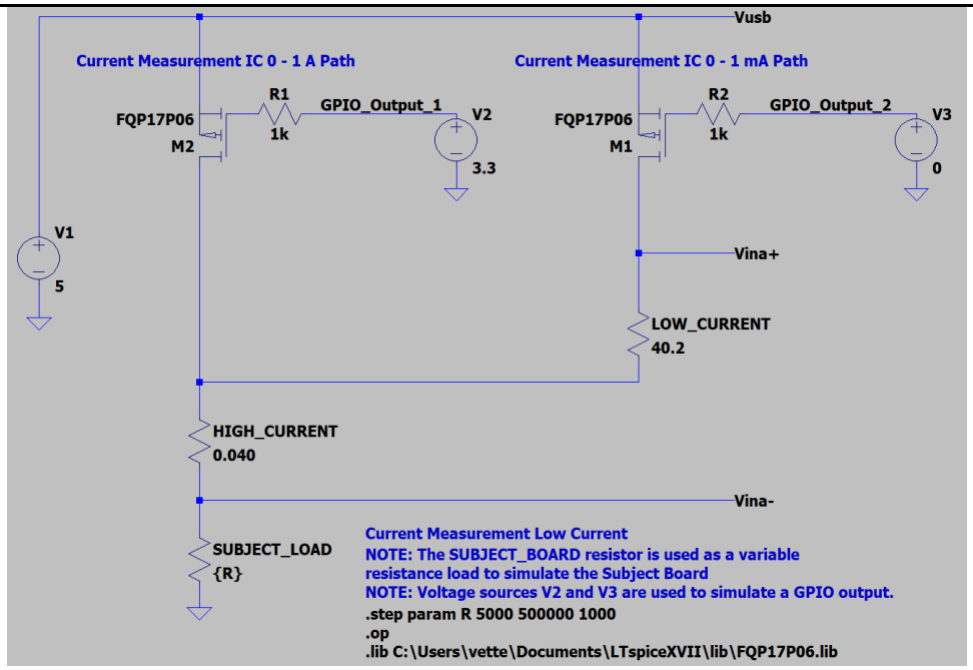


Figure E1: Current IC simulation circuit

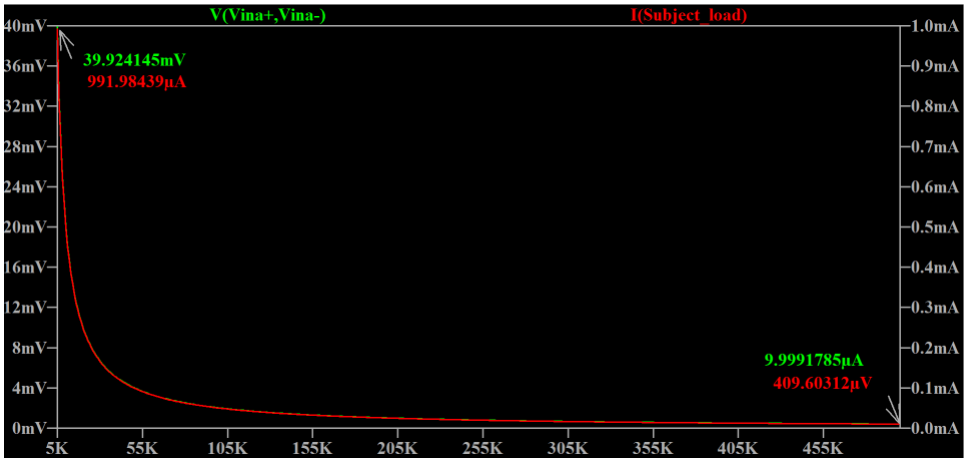


Figure E2: Current IC simulation

From these simulated results, the voltage seen by the INA219BID comes out to 39.92 mV at 991.9 uA and 409.6 uV at 9.999 uA, which aligns with calculations made within 5% error.

Appendix F - Software Setup

Raspi OS Installation

1. Download Raspbian OS Imager (https://www.raspberrypi.com/software/)
2. Insert and locate SD card on computer
3. Install Recommended Raspberry Pi OS (w/ Software) to SD card.
4. Boot-up Raspi
5. Username: microdev | Password: microdev
6. Connect to Internet
7. Terminal: git clone https://github.com/hancheyn/MicroDev.git
8. Terminal: cd MicroDev
9. Terminal: chmod 777 setup.sh
10. Terminal: ./setup.sh
11. File Manager > Edit > Preferences > Volume Management >
[uncheck] Show available options for removable media when they are inserted
12. Raspberry Pi Configuration > [disable] Screen Blanking

Arduino CLI Installation & Application

Start up raspberry pi using an hdmi monitor, keyboard, and mouse.

Download in Add/Remove Software:

1. Raspberry OS Start Menu > Preferences > Add/Remove Software
2. Search “arduino”
3. Check AVR development board IDE
4. Check Command Line Tool for compiling Arduino Sketches
5. Click Apply and Wait for Download

Download in Terminal:

1. Terminal: curl -fsSL https://raw.githubusercontent.com/arduino/arduino-cli/master/install.sh | sh
2. Put arduino-cli into a runnable folder so the command will run on its own ($PATH).
3. Terminal Command: arduino-cli core install arduino:avr
4. Terminal Command View Connected Boards: arduino-cli board list
5. Terminal Command Compile .ino example:
   arduino-cli compile -b arduino:avr:uno Blink.ino -e
6. Terminal Command Upload .ino example:
   arduino-cli upload -b arduino:avr:uno -p /dev/ttyACM0 Blink.ino

FLASH TO ARDUINO NOTES:
The Arduino CLI does not automatically create a permanent bit file for uploading. In order to do this, one must add the command ‘-e’ while compiling. This looks like the following: *arduino-cli compile -b arduino:avr:uno Blink.ino -e*.This will create a new folder within the arduino project folder that contains a bit file, “[Project name].ino.with_bootloader.bin”. This should be in a folder with a file path

./build/arduino.avr.uno/. In Terminal, the command *arduino-cli upload -b arduino:avr:uno -p /dev/ttyACM0 -i [binary file path]* will upload the file.

STM32 CLI Installation & Application

Start up raspberry pi using an hdmi monitor, keyboard, and mouse.

Download in Add/Remove Software:

1. Raspberry OS Start Menu > Preferences > Add/Remove Software
2. Search “stlink”
3. Check OpenSource ST-Link tools replacement… (x3)

a. …libstlink1-1.6.1+ds-3
b. …stlink-gui-1.6.1+ds-3
c. …stlink-tools-1.6.1+ds-3

4. Click Apply and Wait for Download

FLASH TO STM32 NOTES:

The STMCubeIDE does not automatically create a permanent bit file for uploading. Since it uses a .elf file (located in the debug folder) this must first be converted to a .bin file before flashing can occur. The .elf file can be converted using the following command from terminal *arm-none-eabi-objcopy -O binary F401RE_T.elf main.bin.* In the previous example, F401RE_T.elf becomes a binary file called main.bin. Flashing can then occur using the command *st-flash write main.bin 0x08000000*.

I2C Setup and Configuration

Start up raspberry pi using an hdmi monitor, keyboard, and mouse.

Configure I2C:

1. Raspberry OS Start Menu > Preferences >Raspberry Pi Configuration
2. Click on the Interfaces tab.
3. Switch On I2C and click OK
4. Restart Raspberry Pi
5. Terminal: sudo i2cdetect -y 1
6. Check that all i2c device addresses are shown while connected to Bigfoot PCB.

Read-Only Configuration

Follow the procedure in the link below to configure a read only file structure for important directories.
https://learn.adafruit.com/read-only-raspberry-pi

References & Helpful Links:
https://www.raspberrypi.com/software/
https://www.arduino.cc/pro/cli
https://arduino.stackexchange.com/questions/54098/arduino-ide-permision-denied-for-upload-ubuntu
https://www.instructables.com/Start-Developing-STM32-on-Linux/
https://blog.podkalicki.com/how-to-compile-and-burn-the-code-to-stm32-chip-on-linux-ubuntu/
https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial/all
https://raspberry-projects.com/pi/programming-in-python/i2c-programming-in-python/using-the-i2c-interface-2

Appendix G - Pin & Power Mode ID to Address Mapping

STM Nucleo Pin Mapping

Below is the mapping of the Littlefoot PCB Multiplexer enable and address to the corresponding pin ID number, seen in Figure 1.

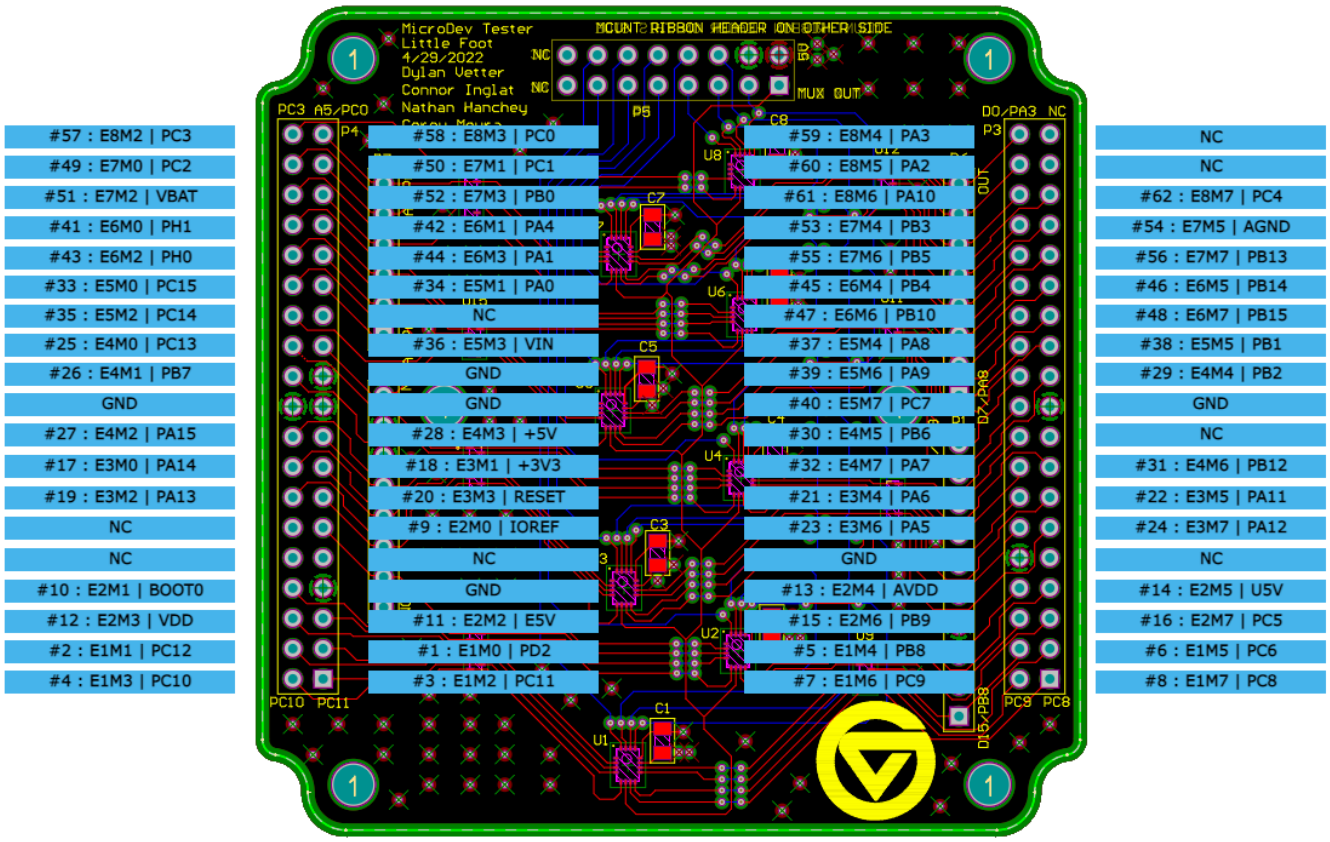


Figure 1: STM32 Nucleo Pin ID Mapping

STM32 GPIO (General Purpose Input Output) Pins

Table 1: STM GPIO Pin Mapping

Pin ID	Mux Address	Enable	Pin Name
1	0	1	PD2
2	1	1	PC12
3	2	1	PC11
4	3	1	PC10
5	4	1	PB8
6	5	1	PC6
7	6	1	PC9
8	7	1	PC8
15	6	2	PB9
16	7	2	PC5
17	0	3	PA14
19	2	3	PA13
21	4	3	PA6
22	5	3	PA11
23	6	3	PA5
24	7	3	PA12
25	0	4	PC13
26	1	4	PB7
27	2	4	PA15
29	4	4	PB2
30	5	4	PB6
31	6	4	PB12
32	7	4	PA7
33	0	5	PC15
34	1	5	PA0
35	2	5	PC14
37	4	5	PA8
38	5	5	PB1
39	6	5	PA9
40	7	5	PC7

41	0	6	PH1
42	1	6	PA4
43	2	6	PH0
44	3	6	PA1
45	4	6	PB4
46	5	6	PB14
47	6	6	PB10
48	7	6	PB15
49	0	7	PC2
50	1	7	PC1
52	3	7	PB0
53	4	7	PB3
55	6	7	PB5
56	7	7	PB13
57	2	8	PC3
58	3	8	PC0
59	4	8	PA3
60	5	8	PA2
61	6	8	PA10
62	7	8	PC4

STM32 Analog Input Pins

Table 2: STM ADC Pin Mapping

Pin ID	Mux Address	Enable	Pin Name
16	7	2	PC5
62	7	8	PC4
38	5	5	PB1
49	0	7	PC2
57	2	8	PC3
34	1	5	PA0
44	3	6	PA1
42	1	6	PA4
52	3	7	PB0
50	1	7	PC1
58	3	8	PC0

STM32 Power Mode IDs

Table 3: STM Power Modes

Power Mode ID	Power Mode Type
1	**Sleep Mode**
2	**Stop Mode**
3	**Standby Mode**

STM32 Wake Up Pins

Pin PA0 (or potentially PA14 if reconfigured)

Arduino Uno Pin Mapping

Below is the mapping of the Littlefoot PCB Multiplexer enable and address to the corresponding pin ID number, seen in Figure 2:

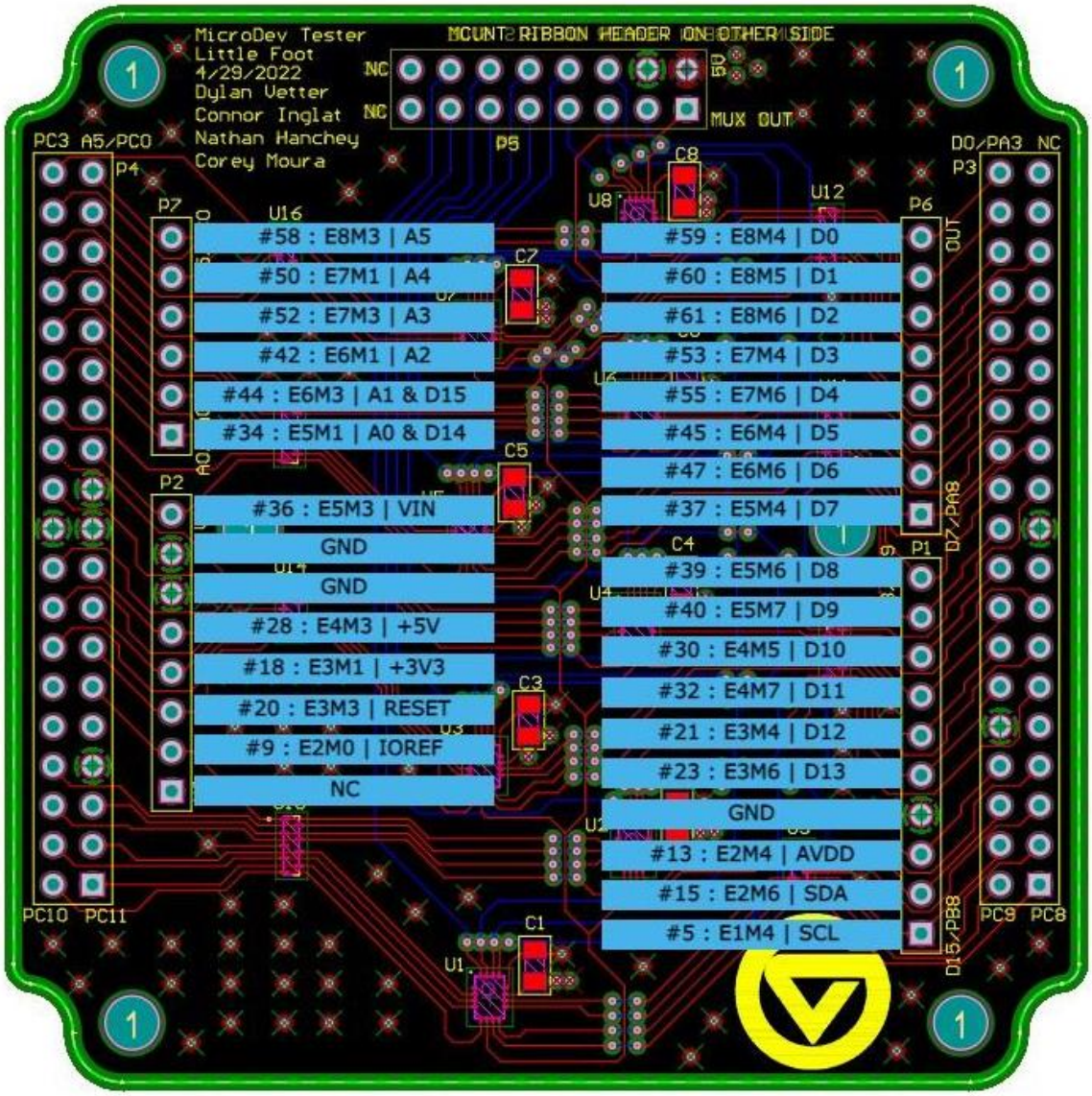


Figure 2: Arduino Uno Pin Mapping

Arduino Uno GPIO (General Purpose Input Output) Pins

Table 4: Arduino GPIO Pin Mapping

Pin ID	Mux Address	Enable	Pin Name
5	4	1	A5/SDA
15	6	2	A4/SCL
21	4	3	D12
23	6	3	D13
30	5	4	D10
32	7	4	D11
37	4	5	D7
39	6	5	D8
40	7	5	D9
45	4	6	D5
47	6	6	D6
53	4	7	D3
55	6	7	D4
59	4	8	D0
60	5	8	D1
61	6	8	D2
34	1	5	A0
44	3	6	A1
42	1	6	A2
52	3	7	A3
50	1	7	A4
58	3	8	A5

Arduino Uno Analog Input Pins

Table 5: Arduino Uno ADC Pin Mapping

Pin ID	Mux Address	Enable	Pin Name
34	1	5	A0
44	3	6	A1
42	1	6	A2
52	3	7	A3
50	1	7	A4
58	3	8	A5

Arduino Uno Power Modes ID

Table 6: Arduino Uno Power Modes

Power Mode ID	Power Mode Type
1	**Sleep Mode Idle**
2	**Sleep Mode ADC**
3	**Sleep Mode Power Save**
4	**Sleep Mode EXT Standby**
5	**Sleep Mode Standby**
6	**Sleep Mode Power Down**

Arduino Uno Wake Up Pins

Pin D2 or D3

Appendix H - Bigfoot Function Prototypes

`def set_volt(float: vout) -> None:`

- Description:
  - This function sets the voltage that will be outputted by the digital to analog converter.
- Arguments:
  - float: vout - Voltage that DAC will be set.
- Returns:
  - None

`def set_mux_add(int: state, int: enable, int: add) -> None:`

- Description:
  - This function resets the multiplexers to a default state and enables one to connect and isolate a pin on the subject board to the i2c sensors on the Bigfoot PCB.
- Arguments:
  - int: state - When the state is 0 then all multiplexers are disabled and the address signals are set low.
  - int: enable - Indicates the multiplexer that will be enabled.
  - int : add - Indicates the address that is to be enabled.
- Returns:
  - None

`def rpi_i2c_config() -> None:`

- Description:
  - This function configures the i2c peripheral.
- Arguments:
  - N/A
- Returns:
  - None

`def rpi_i2c_adc() -> float:`

- Description:
  - This function captures the voltage signal on the multiplexer out line.
- Arguments:
  - N/A
- Returns:
  - float - Voltage measurement in volts.

`def rpi_i2c_dac() -> None:`

- Description:

- ➢ This function generates a voltage signal on the multiplexer out line.

❖ Arguments:
- ➢ N/A

❖ Returns:
- ➢ None

---

def rpi_i2c_ina219(int: shunt) -> float:

❖ Description:
- ➢ This function returns the current in mA from the INA219 sensor.

❖ Arguments:
- ➢ int: shunt - Indicates whether the shunt is for high currents or low currents with 0 or 1.

❖ Returns:
- ➢ float - Current measurement in mA.

---

def high_current(int: state) -> None:

❖ Description:
- ➢ This function enables the high current shunt resistance on the Bigfoot PCB.

❖ Arguments:
- ➢ int: state - Indicates whether the switch is enabled or disabled with 0 or 1 respectively.

❖ Returns:
- ➢ None

---

def low_current(int: state) -> None:

❖ Description:
- ➢ This function enables the low current shunt resistance on the Bigfoot PCB.

❖ Arguments:
- ➢ int: state - Indicates whether the switch is enabled or disabled with 0 or 1 respectively.

❖ Returns:
- ➢ None

---

def dac_enable(int: state) -> None:

❖ Description:
- ➢ This function enables the DAC peripheral on the Bigfoot PCB.

❖ Arguments:
- ➢ int: state - Indicates whether the switch is enabled or disabled with 1 or 0 respectively.

❖ Returns:
- ➢ None

---

def adc_load(int: state) -> None:

- Description:
  - This function enables the load resistance in line with the ADC on the Bigfoot PCB.
- Arguments:
  - int: state - Indicates whether the switch is enabled or disabled with 1 or 0 respectively.
- Returns:
  - None

---

def adc_enable(int: state) -> None:

- Description:
  - This function enables the ADC peripheral on the Bigfoot PCB.
- Arguments:
  - int: state - Indicates whether the switch is enabled or disabled with 1 or 0 respectively.
- Returns:
  - None

---

def signal_handler(int: sig, int: frame) -> None:

- Description:
  - This function is the signal handler for an interrupt.
- Arguments:
  - int: sig - placeholder variable
  - int: frame - placeholder variable
- Returns:
  - None

---

def b1_release(int: channel) -> None:

- Description:
  - This function handles the release of button 1 after the interrupt for the GPIO is triggered.
- Arguments:
  - int: channel- placeholder variable
- Returns:
  - None

---

def b2_release(int: channel) -> None:

- Description:
  - This function handles the release of button 2 after the interrupt for the GPIO is triggered.
- Arguments:
  - int: channel- placeholder variable
- Returns:
  - None

---

def b3_release(int: channel) -> None:

- Description:
  - This function handles the release of button 3 after the interrupt for the GPIO is triggered.
- Arguments:
  - int: channel- placeholder variable
- Returns:
  - None

---

def b1_enable() -> None:

- Description:
  - This function enables button 1 by configuring the input and interrupt. It also resets the global variable that indicates the button has been pressed.
- Arguments:
  - N/A
- Returns:
  - None

---

def b2_enable() -> None:

- Description:
  - This function enables button 2 by configuring the input and interrupt. It also resets the global variable that indicates the button has been pressed.
- Arguments:
  - N/A
- Returns:
  - None

---

def b3_enable() -> None:

- Description:
  - This function enables button 3 by configuring the input and interrupt. It also resets the global variable that indicates the button has been pressed.
- Arguments:
  - N/A
- Returns:
  - None

---

def b1_disable() -> None:

- Description:
  - This function disables the button presses by resetting the bit for button 1.
- Arguments:
  - N/A
- Returns:

- ➢ None

---

def b2_disable() -> None:

- ❖ Description:
  - ➢ This function disables the button presses by resetting the bit for button 2.
- ❖ Arguments:
  - ➢ N/A
- ❖ Returns:
  - ➢ None

---

def b3_disable() -> None:

- ❖ Description:
  - ➢ This function disables the button presses by resetting the bit for button 3.
- ❖ Arguments:
  - ➢ N/A
- ❖ Returns:
  - ➢ None

---

def get_button_state() -> int:

- ❖ Description:
  - ➢ This function finds the value representing the state of the buttons.
- ❖ Arguments:
  - ➢ N/A
- ❖ Returns:
  - ➢ int - unsigned value that indicates the state of the buttons through the first three bits of the value.

---

def pollButton() -> String:

- ❖ Description:
  - ➢ This function polls the button inputs to decipher whether a button has been pressed.
- ❖ Arguments:
  - ➢ N/A
- ❖ Returns:
  - ➢ String - the string indicates which button has been pressed after polling the buttons.

Appendix I - View Function Prototypes

`def setStandbyScreen() -> None:`

- Description:
  - This function updates the screen to show a message that indicates a subject board needs to be connected.
- Arguments:
  - N/A
- Returns:
  - None

`def setStartScreen(string: board_type) -> None:`

- Description:
  - This function updates the screen to show a message that indicates the test cycle is ready to begin.
- Arguments:
  - String: board_type - The board type string indicates which board has been detected to show on screen.
- Returns:
  - None

`def setMessageScreen(string: message) -> None:`

- Description:
  - This function updates the screen to show the message passed as a parameter.
- Arguments:
  - String: message - Message to display on screen.
- Returns:
  - None

`def setFlashScreen() -> None:`

- Description:

- This function updates the screen to show a message that indicates the subject board is being flashed with the testing program.

❖ Arguments:
- N/A

❖ Returns:
- None

---

def setRunningScreen(int: percent) -> None:

❖ Description:
- This function updates the screen to show a message that indicates the percentage of tests completed.

❖ Arguments:
- int: percent - Percentage of tests that have been completed.

❖ Returns:
- None

---

def setResultsScreen(string array: pass_array) -> None:

❖ Description:
- This function updates the screen to show a message that shows the test cycle general results.

❖ Arguments:
- String array: pass_array - This array of Strings contains the results of the testing cycle.

❖ Returns:
- None

---

def setDetailedTestScreen(string array: detailed_report) -> None:

❖ Description:
- This function updates the screen to show a message that shows the test cycle detailed results. It also reconfigures the press buttons to scroll through text and blocks the program until the middle button is pressed.

❖ Arguments:
- String array: detailed_report - This array of Strings contains the detailed results of the testing cycle.

❖ Returns:
- None

---

def setSaveScreen(string: save_condition) -> None:

❖ Description:
- This function updates the screen to show a message that indicates whether or not the detailed results successfully saved.

❖ Arguments:
- String: save_condition - String indicates whether the save to a flash drive was successful or not.

❖ Returns:
- None

---

def setShutdownScreen() -> None:

- Description:
  - This function updates the screen to show a message that indicates a shutdown procedure has begun.
- Arguments:
  - N/A
- Returns:
  - None

```
def setRemovalScreen() -> None:
```

- Description:
  - This function updates the screen to show a message that indicates the subject board should be removed.
- Arguments:
  - N/A
- Returns:
  - None

Appendix J - Model Function Prototypes

```
def open_serial( ) -> object:
```

- Description:
  - Opens a serial port to the subject board.
- Arguments:
  - **N/A**
- Returns:
  - Object - Serial port information

```
def close_serial( ) -> None:
```

- Description:
  - This function closes a serial port.
- Arguments:
  - N/A
- Returns:
  - None

```
def subject_write( string: str_write, object: ser ) -> None:
```

- Description:
  - This function writes a string to a serial port object.
- Arguments:
  - N/A
- Returns:
  - None

---

def subject_read(object: ser) -> byte array:

- Description:
  - This function reads from a serial port object.
- Arguments:
  - N/A
- Returns:
  - byte array - Data packet read from subject board.

---

def crc_decode(byte array: value, int: out_type) -> int:

- Description:
  - This function decodes a byte array from a serial port object.
- Arguments:
  - N/A
- Returns:
  - int - decoded value for either pin, test, or results values.

---

def crc_encode(int: test, int: pin, int: instruction) -> byte array:

- Description:
  - This function encodes a message before sending it through serial communication.
- Arguments:
  - N/A
- Returns:
  - byte array - encoded data packet to send through serial communication.

---

def serial_setup( ) -> int:

- Description:
  - This function is a check to confirm the serial communication is running properly.
- Arguments:
  - N/A
- Returns:
  - int - This value is the received data from the serial communication.

---

def run_subject_test(int: pin, int: enable, int: address, int: test, int: instruction, object: ser) -> float:

- Description:
  - This function identifies the test and runs the unique procedure for that test.
- Arguments:
  - int: pin - Pin identification number.
  - int: enable - Multiplexer enable for specific pin.
  - int: address - Multiplexer address for specified pin.

- int: test - Test identification number.
- int: instruction - instruction used to configure a test.
- Object: ser - Serial port information.

❖ Returns:

- float - Measured voltage value in volts.

---

def run_gpio_output_test(int: pin, int: enable, int: address, int: instruction, object: ser) -> float:

❖ Description:

- This function contains the procedure for the GPIO output test.

❖ Arguments:

- int: pin - Pin identification number.
- int: enable - Multiplexer enable for specific pin.
- int: address - Multiplexer address for specified pin.
- int: instruction - instruction used to configure a test.
- Object: ser - Serial port information.

❖ Returns:

- float - Measured voltage value in volts.

---

def run_gpio_output_loading_test(int: pin, int: enable, int: address, int: instruction, object: ser) -> float:

❖ Description:

- This function contains the procedure for the GPIO output loading test.

❖ Arguments:

- int: pin - Pin identification number.
- int: enable - Multiplexer enable for specific pin.
- int: address - Multiplexer address for specified pin.
- int: instruction - instruction used to configure a test.
- Object: ser - Serial port information.

❖ Returns:

- float - Measured voltage value in volts.

---

def run_gpio_input_pull_up_test(int: pin, int: enable, int: address, int: instruction, object: ser) -> float:

❖ Description:

- This function contains the procedure for the GPIO input pullup test.

❖ Arguments:

- int: pin - Pin identification number.
- int: enable - Multiplexer enable for specific pin.
- int: address - Multiplexer address for specified pin.
- int: instruction - instruction used to configure a test.
- Object: ser - Serial port information.

❖ Returns:

- float - Measured voltage value in volts.

`def run_gpio_input_pull_down_test( int: pin, int: enable, int: address, int: instruction, object: ser ) -> float:`

- Description:
  - This function contains the procedure for the GPIO input pulldown test.
- Arguments:
  - int: pin - Pin identification number.
  - int: enable - Multiplexer enable for specific pin.
  - int: address - Multiplexer address for specified pin.
  - int: instruction - instruction used to configure a test.
  - Object: ser - Serial port information.
- Returns:
  - float - Measured voltage value in volts.

`def run_gpio_input_logic_level_test( int: pin, int: enable, int: address, int: instruction, object: ser ) -> int:`

- Description:
  - This function contains the procedure for the GPIO input logic level test.
- Arguments:
  - int: pin - Pin identification number.
  - int: enable - Multiplexer enable for specific pin.
  - int: address - Multiplexer address for specified pin.
  - int: instruction - instruction used to configure a test.
  - Object: ser - Serial port information.
- Returns:
  - int - 1 or 0 indicating high or low voltage level.

`def run_adc_test( int: pin, int: enable, int: address, int: instruction, object: ser ) -> float:`

- Description:
  - This function contains the procedure for the ADC test.
- Arguments:
  - int: pin - Pin identification number.
  - int: enable - Multiplexer enable for specific pin.
  - int: address - Multiplexer address for specified pin.
  - int: instruction - instruction used to configure a test.
  - Object: ser - Serial port information.
- Returns:
  - float - Voltage of subject board ADC.

`def run_power_mode_test( int: pin, int: instruction, object: ser ) -> float:`

- Description:
  - This function contains the procedure for the power mode test.
- Arguments:

- ➢ int: pin - Pin identification number.
- ➢ int: instruction - instruction used to configure a test.
- ➢ Object: ser - Serial port information.

❖ Returns:

- ➢ float

---

def run_wakeup_test(int: pin, int: enable, int: address, int: instruction) -> float:

❖ Description:

- ➢ This function contains the procedure for the wakeup test.

❖ Arguments:

- ➢ int: pin - Pin identification number.
- ➢ int: enable - Multiplexer enable for specific pin.
- ➢ int: address - Multiplexer address for specified pin.
- ➢ int: instruction - instruction used to configure a test.

❖ Returns:

- ➢ float - Measured current in mA.

---

def current_read( ) -> float:

❖ Description:

- ➢ This function contains the procedure for the current read test.

❖ Arguments:

- ➢ N/A

❖ Returns:

- ➢ float - Measured current in mA.

---

def subject_flash(string: board) -> boolean:

❖ Description:

- ➢ This function flashes the unique software based on the subject board that was detected, to that subject board.

❖ Arguments:

- ➢ String: board - Indicates type of subject board detected.

❖ Returns:

- ➢ Boolean - Indicates successful flash when True.

---

def board_list( ) -> string:

❖ Description:

- ➢ This function detects the subject board when it is connected through usb.

❖ Arguments:

- ➢ N/A

❖ Returns:

- ➢ String - Indicates board that was detected.

---

def board_wait( ) -> string:

- Description:
  - This function holds in a loop until a subject board is detected.
- Arguments:
  - N/A
- Returns:
  - String - Indicates board that was detected.

---

def usb_list( ) -> string:

- Description:
  - This function finds and lists the usb flash drives that are connected.
- Arguments:
  - N/A
- Returns:
  - String - Filepath for usb drive.

---

def usb_save(string array: detailed_array) -> string:

- Description:
  - This function saves the detailed results files to an attached USB flash drive.
- Arguments:
  - String array: detailed_array - Detailed results for every pin and test in a test cycle.
- Returns:
  - String - Indicates whether results saved to usb drive.

---

def shutdown( ) -> None:

- Description:
  - This function sends a shutdown instruction to the terminal.
- Arguments:
  - N/A
- Returns:
  - None

---

def check_5V( ) -> float:

- Description:
  - This function runs a test on the 5V output pin of the subject board and measures it with the Bigfoot ADC.
- Arguments:
  - N/A
- Returns:
  - float - 5V pin measured voltage

```
def check_3V3( ) -> float:
```

- Description:
  - This function runs a test on the 3V3 output pin of the subject board and measures it with the Bigfoot ADC.
- Arguments:
  - N/A
- Returns:
  - float - 3V3 pin measured voltage

Appendix K - Controller Function Prototypes

`def serial_check() -> bool:`

- Description:
  - This function confirms that serial communication with the subject board is established.
- Arguments:
  - N/A
- Returns:
  - Boolean - True is a successful serial check and False is not.

`def test_config_file(string: _board_type) -> string array:`

- Description:
  - This function returns the test cycle configurations that indicate which tests and pins are to be tested in a test cycle.
- Arguments:
  - String: _board_type - Indicates which subject board has been detected.
- Returns:
  - String array - Configuration for test cycle

`def threshold_config_file(string: _board_type) -> string array:`

- Description:
  - This function returns the threshold configurations for a given subject board.
- Arguments:
  - String: _board_type - Indicates which subject board has been detected
- Returns:
  - String array - Configurations for threshold values

`def pinmap_array() -> key array:`

- Description:
  - This function returns the pin mapping for the desired subject board.
- Arguments:
  - N/A
- Returns:
  - Key array - Array that contains the pin mapping of Pin IDs to Pin names.

`def subject_logic(string: _board_type) -> float:`

- Description:
  - This function returns the logic required for a given subject board.
- Arguments:

- String: _board_type - Indicates which subject board has been detected

❖ Returns:

- float - high logic voltage

---

def subject_pin_voltages(string: _board_type) -> bool:

❖ Description:

- This function tests whether the 3V3 and 5V output pins of the development board are connected.

❖ Arguments:

- String: _board_type - Indicates which subject board has been detected

❖ Returns:

- bool - True indicates voltages pass threshold levels.

---

def subject_test(int: t, int: p, int: a, int: e, string: board, object: _ser) -> None:

❖ Description:

- This function runs one line of the test configurations that performs a test on the specified pin.

❖ Arguments:

- int: t - Indicates the test ID.
- int: p - Indicates the pin ID.
- int: a - Indicates the address that needs to be sent to the multiplexer.
- int: e - Indicates which multiplexer must be enabled.
- String: board - Indicates which subject board has been detected
- Object: _ser -The serial port that has been opened.

❖ Returns:

- None

---

def resets_menu(bool: redo, string array: pass_array, string array: detailed_array) -> None:

❖ Description:

- This function runs after a test cycle to display the results.

❖ Arguments:

- Boolean: redo - Boolean value to continue or skip results menu.
- String array: pass_array - The general results for each test.
- String array: detailed_array- The detailed results for each pin.

❖ Returns:

- None

Appendix L - Subject Board Function Prototypes

```
void configure_output(unsigned int pin, unsigned int logic)
```

- Description:
  - Sets a GPIO pin to a high or low logic output.
- Arguments:
  - unsigned int pin - Represents the pin identification value read from the serial data packet.
  - unsigned int logic - Represents the high or low logic needed for the given test.
- Returns:
  - None

```
int configure_input(unsigned int pin)
```

- Description:
  - Sets a given GPIO pin as a floating input.
- Arguments:
  - unsigned int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
void configure_input_pullup(unsigned int pin)
```

- Description:
  - Sets a given GPIO pin as an input pullup.
- Arguments:
  - unsigned int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
void configure_input_pulldown(unsigned int pin)
```

- Description:
  - Sets a given GPIO pin as an input pulldown, if applicable.
- Arguments:
  - unsigned int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
int configure_analog_input(unsigned int pin)
```

- Description:
  - Sets a given Analog pin as an ADC input.

- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
void reset_pins()
```

- Description:
  - Handles reconfiguration of all GPIO pins to a default state, floating input.
- Arguments:
  - None
- Returns:
  - None

```
int command_read(unsigned char data[])
```

- Description:
  - Handles the facade tests between subject board and Raspberry Pi during testing of GPIO pull-up inputs, while unloaded.
- Arguments:
  - unsigned char data[] - Array of the 3 byte data packet containing results of a test to send through serial communication.
- Returns:
  - None

```
int command_write(unsigned int pin, unsigned int result, unsigned int test)
```

- Description:
  - Handles facade
- Arguments:
  - unsigned char data[] - Array of the 3 byte data packet containing results of a test to send through serial communication.
- Returns:
  - None

```
int crc_encode(unsigned char data[], unsigned int pin, unsigned int result, unsigned int test)
```

- Description:
  - Handles the encoding for the data packet to serial.
- Arguments:
  - unsigned char data[] - Array of the 3 byte data packet containing results of a test to send through serial communication.
- Returns:
  - None

```
int crc_decode(unsigned char data[])
```

- Description:
  - Handles the decoding of the data packet from serial.
- Arguments:
  - unsigned char data[] - Array of the 3 byte data packet for running a test on a pin or sleep mode..
- Returns:
  - None

```
int analogRead(int pin)
```

- Description:
  - Handles the process to obtain an analog to digital voltage capture.
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
int digitalRead(int pin)
```

- Description:
  - Handles the process to obtain an input logic read.
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
void analogWrite(int pin, int value)
```

- Description:
  - Handles the process to configure a PWM or DAC write, if required.
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
void digitalWrite(int pin, int logic)
```

- Description:
  - Handles the process to configure a digital signal on an output pin.
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:

- None

```
void pinMode(int pin, int mode)
```

- Description:
  - Handles the pin configurations for different types of inputs and output.
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
  - int mode - Represents what type of configuration is required.
- Returns:
  - None

```
void configure_sleep_mode(unsigned int sleepmode, unsigned int interruptPin)
```

- Description:
  - Handles the configuration of the sleep modes. Setting the interrupt pin is not yet fully supported.
- Arguments:
  - unsigned int sleepmode - Represents the sleep mode that will be configured.
  - unsigned int interruptPin - Represents the interrupt pin to set for wake up from sleep mode.
- Returns:
  - None

```
void run_tests(unsigned char data[])
```

- Description:
  - Handles
- Arguments:
  - int pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - None

```
struct pin pin_set(uint32_t pin, uint32_t reg_values, …, uint8_t pinid)
```

- Description:
  - Handles creating an object for a single pin that contains the required register values for configuration.
- Arguments:
  - uint_32_t pin - Represents the pin register value required to configure I/O pin.
  - uint_32_t reg_values - Represents other important register values for pin configuration as required by software developers.
  - uint_8_t pin - Represents the pin identification value read from the serial data packet.
- Returns:
  - struct pin

```
void init_pins(struct pin pins[])
```

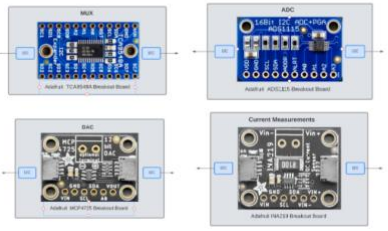
- Description:
  - Handles the initialization of the pins structure array which maps the pin identification integer to the important register values of that pin.
- Arguments:
  - struct pin pins[] - This struct array contains the important register values of each I/O pin on a subject board. The index value is made to line up with the pin ID value given to a specific pin.
- Returns:
  - None (alters struct pin as output)

Appendix M - Concept Selection Phase

Appendix M contains the concepts that were generated when brainstorming ideas during the Empathize phase of the project's development.

Breakout Board Design of ICs

Breakout boards will be used in measuring the voltage and current coming from the subject board, as well as generating signals to send to the subject board.



Implements:
- Sensory Inputs
- Voltage Output
- Pin Isolation

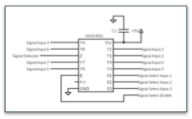
Pros:
- Robust design
- Modularity
- No PCB Design
- Scored 4.14/5 on Morphological Matrix

Cons:
- No control over circuit design
- Greater Cost
- Larger Size
- More difficult to fabricate

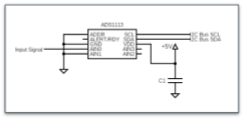
Custom PCB of IC's (ADC, DAC, Current)

Below is a general implementation of these ICs.

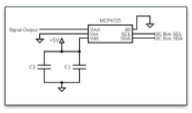
Multiplexer



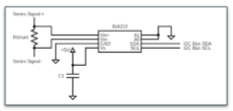
Voltage Measurement



DAC



Current Measurement

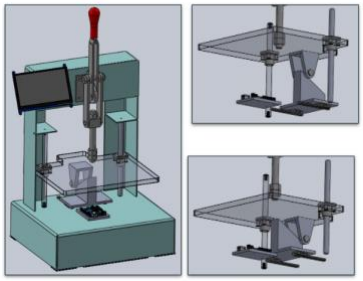


Implements:
- Sensory Inputs
- Voltage Outputs

Pros:
- Smaller Footprint
- Control Over Circuit Design
- Lower Cost

Cons:
- "Reinventing the wheel"
- Lacks Modularity
- Design complexity
- Scored 3.45/5 on Morphological Matrix

Enclosure Concept 1

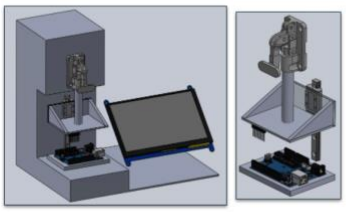


Implements:
- Sheet metal frame
- Hold down toggle clamp initiated (detents)
- Rail and carriages for horizontal motion
- Linear Bearing and rails for vertical motion

Pros:
- Extremely robust
- Removes operator error when interfacing

Cons:
- Very high cost
- Reduced production quantity based on budget
- Design complexity is high
- Increased manufacturing time
- Large footprint

Enclosure Concept 2

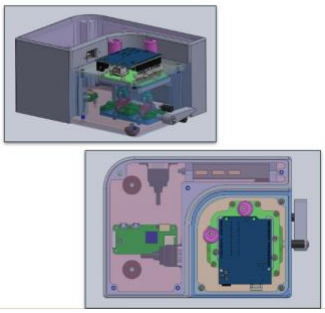


Implements:
- Sheet metal frame
- 3D printed electrical enclosure
- Toggle clamp initiated interface

Pros:
- Robust design
- Medium weight
- Similar design to industrial applications

Cons:
- Medium - high cost
- Many production processes
- Design complexity

Enclosure Concept 3



Implements:
- 2 piece 3D printed design
- Mostly 3D printed components
- Mechanical ejection system

Pros:
- Light weight
- Low cost
- More portable
- Ease of production (others can reproduce)
- Low profile

Cons:
- Not as robust

Software Design of GUI Concept 1 (Touch Screen)

This GUI concept incorporates various tabs used on the same window to convey information and takes basic inputs from the user. This concept takes user input through the touch screen itself.

Implements:
- Touch screen display interface

Pros:
- Easier to view displayed information
- Touch screen inputs get in the way still

Cons:
- Displayed information is divided into separate tabs
- Increased complexity

Software Design of GUI Concept 2 (External Push Button)

This GUI concept incorporates various tabs used on the same window to convey information and takes basic inputs from the user. However, this concept frees up screen space through using push buttons to toggle through the tabs.

Implements:
- Touch screen display interface

Pros:
- Easier to view displayed information
- Clearest viewability

Cons:
- Displayed information is divided into separate tabs
- Most complex concept

Software Design of GUI Concept 3

This GUI concept incorporates no tabs used on a single window to convey information and takes basic inputs from the user.

Implements:
- Touch screen display interface

Pros:
- All displayed information is visible at once
- Decreased complexity

Cons:
- Harder to view displayed information

Appendix N - Scoping Document

GVSU Capstone Project 2022

Sponsor: GVSU

Project: MicroDev Tester

Project Specifications and Scope of Work Approval – Rev.1

Dr. Karl Brakora – GVSU Sponsor Contact

Signature Karl Brakora Date 2/18/2022

Dr. Nicholas Baine – GVSU Team Advisor

Signature Nicholas A. Baine Date 2/18/2022

Corey Moura– Team Captain

Signature Corey Moura Date 4/9/2022

Nathan Hanchey– Sponsor and GVSU Advisor Point of Contact

Signature Nathan Hanchey Date 02/22/2022

Connor Inglat– Treasurer

Signature Connor Inglat Date 02/22/2022

Dylan Vetter – Secretary

Signature Dylan Vetter Date 02/22/2022

Statement of Purpose:

There are several engineering courses where students are required to program and build hardware circuits with their microcontroller. Hardware connections made incorrectly can destroy or damage individual pins and in some cases the entire board. Additionally, some boards are damaged before being used by the student.

Students and professors often have the need to identify whether unusual behavior is hardware or software related in embedded system projects. Oftentimes, it can be quite cumbersome to diagnose hardware related issues that may occur on a microcontroller, such as damaged GPIO pins, voltage regulators, and other critical hardware peripherals. This presents a need to find a more expedient way to diagnose these issues when many students may require help.

This project will solve this problem by checking the functionality of the Arduino Uno and STM microcontroller hardware through a series of hardware tests. This will allow students and professors to confirm hardware faults or rule out any hardware errors thought to be causing issues. If hardware is not behaving properly, the device created by this project will display the hardware failures on a Graphical User Interface. Otherwise, the student or professor can assume their issues to be coming from software.

Definition of Terms:

Device - One testing system unit.
Arduino - Reference to Arduino Uno board under test.
STM - Reference to STM32 board under test.
Subject Board - Reference to either of the possible boards under test.

Device System Description:

The device will have a switch that toggles power supplied to internal components. The subject board will be inserted onto a fixture made to the specifications of the subject board. A USB connection will be made from the device to the subject board by the user. Once the USB connection is made to the subject board a start button will appear on the GUI to begin testing. The operator can then begin the test by pressing the start button.

The device's electrical pin interface establishes a connection to each of the subject board's pins which leads into the test circuit and circuit protection PCB. The PCB includes the analog circuits and external peripherals required for the completion of the subject board testing. The PCB will be interfaced to the microprocessor unit using serial communication or a designated type of digital communication. When a test has completed, the detected hardware failures will be displayed to the GUI. The user can access a more detailed report of the test by pressing a labeled button on the GUI. The operator can upload the results via removable media, to which the test results will be automatically saved. This media will also contain a configuration file to allow the operator to adjust each test's pass/fail threshold from their default values.

Functional Requirements

Desktop development board testers will test for damaged hardware on the Arduino Uno and STM32 microcontrollers. The device will allow the user to place an STM or Arduino board onto a fixture to be tested. Once the test is finished, the user can view the results.

Physical Requirements:

1.00	The device **must fit on a desktop**
1.01	The device must be **portable**
1.02	The device **must be physically assembled into a single piece**
1.03	The device must be **simple to load and unload without causing damage to the subject board**
1.04	The device **must have an enclosure that can withstand the rigors of daily use**

Hardware Requirements:

2.00	The device must utilize a physical **switch to control main power**
2.01	The device must be powered from a **120V AC wall outlet**
2.02	The device must implement **circuit protection to avoid damage** to its hardware
2.03	The device must **only require the subject board** from the operator for conduct testing
2.04	The device must have a **servicable hardware interface** to the subject board
2.05	The device must **accurately measure** the subject board **voltages** under test
2.06	The device must **accurately measure** the subject board **currents** under test

Testing Requirements:

3.00	The device must test and record the **voltage supplied** from the subject board's **5V and 3.3V output pins**
3.01	The device must test and record the **current supplied** from the subject board's **5V and 3.3V output pins**
3.02	The device must test and record each GPIO pin's output voltage under **load sourcing**
3.03	The device must test and record each GPIO pin's internal resistor value
3.04	The device must test and record each GPIO pin's internally resistive **pull-up voltage while unloaded**

3.05	The device must test and record each GPIO pin's internally resistive **pull-down voltage while unloaded**
3.06	The device must test and record the GPIO **input pin logic thresholds** of the subject board
3.07	The device must test and record the **accuracy** of the subject board's **ADC pins and channels**
3.08	The device must test and record **all of the pins** capable of **GPIO** wakeups from **sleep mode**
3.09	The device must test all **sleep modes and power modes**

Software Requirements:

4.00	The device must **identify the subject board being tested**
4.01	The GUI implemented on the device must be **intuitive to use**
4.02	The device must allow the user to **start** a **test through** its **GUI**
4.03	The device's GUI must **provide a detailed report** of the subject board's test
4.04	The device's software must **allow configurable thresholds for tests**
4.05	The software developed for the device must be **appropriately documented**

STM Requirements:

5.00	The device **production quantity must be 5**

Arduino Requirements:

6.00	The device **production quantity must be 5**

System Specifications

Physical Specifications:

1.00.1	The device must have overall dimensions not exceeding 1' x 1' x 1'	Required	Measurement
1.01.1	The device must weigh less than 10 lbs	Required	Measurement
1.01.2	The device must not need additional calibration when moved	Required	Inspection
1.02.1	The device must not have unattached or unsecured components	Required	Inspection
1.03.1	The device must use two alignment posts to aid in loading and prevent damage	Required	Inspection
1.03.2	The device must use a mechanism to easily and safely remove the subject board without causing damage	Required	Inspection
1.04.1	The device must have an infill of at least 30% for any 3D printed parts	Required	Inspection

Hardware Specifications:

2.00.1	The device must use a toggle switch to control the main power supply	Required	Measurement
2.01.1	The device must use a 120V AC	Required	Measurement
2.02.1	The device must utilize overvoltage / surge protection IC's for circuits that have potential to experience overvoltage	Required	Measurement
2.04.1	Must have a header that connects to the i/o pins of the subject board	Required	Measurement
2.05.1	The device must implement hardware that measures voltage with an **accuracy of 0.1V within the range 0 to 5.5 V** unless otherwise specified	Required	Measurement
2.06.1	The device must implement hardware that measures current with an **accuracy of 10 uA within the range 0 to 1000uA** (subject to change based on IC)	Required	Measurement
2.06.2	The device must implement hardware that measures current with an **accuracy of 10 mA within the range 0 to 1 A** (subject to change based on IC)	Required	Measurement

Testing Specifications:

3.00.1	The device must test that the subject board’s supply voltage is within a configurable range based on manufacturers specifications	Required	Unit Test & Integration
3.01.1	The device must test that current remains below a configurable threshold based on manufacturers specifications	Required	Unit Test & Integration
3.02.1	The device must test that the pin's output voltage remains above a configurable threshold based on half the manufacturers specified pin’s maximum rated current	Required	Unit Test & Integration
3.03.1	The device must test that the calculated internal resistance of each GPIO pin is within the manufacturers specified range	Required	Unit Test & Integration
3.04.1	The device must measure and test each GPIO pin's voltage while configured as pull-up to be within a configurable range based on the manufacturers specifications	Required	Unit Test & Integration
3.05.1	The device must measure and test each GPIO pin's voltage while configured as pull-down to be within a configurable range based on manufacturers specifications	Required	Unit Test & Integration
3.06.1	The device must test that logic low/high is produced when given a voltage within their threshold based on the manufacturers specifications	Required	Unit Test & Integration
3.07.1	The device must generate voltages based on the manufacturers specifications while testing and measuring the subject board's ADC pin’s input	Required	Unit Test & Integration
3.08.1	The device must generate a signal to wake the subject board	Required	Measurement
3.08.2	The device must test that the subject board's GPIO pins can exit sleep mode	Required	Unit Test & Integration
3.09.1	The device test the subject board's current draw during sleep and power modes to be below a configurable threshold based on the manufacturers specifications	Required	Measurement

Software Specifications:

4.00.1	The device's software must read and record the subject board's identification information found in the Device Descriptor Table	Required	Inspection
4.01.1	Any interactive icons on the device's GUI must be labeled	Required	Inspection
4.02.1	The device's GUI must display a "begin test" button once the subject board is properly loaded	Required	Inspection
4.03.1	The device's GUI must display a report of the failures of the subject board at the conclusion of testing	Required	Inspection
4.03.2	The device must report if an error occurred in the execution of a test	Required	Inspection
4.04.1	The device's software must **use removable media to load a configuration file**	Required	Inspection
4.05.1	The device's software must be commented	Required	Inspection
4.05.2	The device's software must be documented with a manual	Required	Inspection

STM32 Specifications:

5.00.1	There must be 5 total STM testing devices produced	Required	Inspection

Arduino Specifications:

6.00.1	There must be 5 total Arduino testing devices produced	Required	Inspection

Deliverables:

- 5 MicroDev Arduino Testing Devices
- 5 MicroDev STM Testing Devices
- User Manual
- Device Design Development Documentation
- Documented Validation Testing to Verify Device Performance

Test Methods:

Visual Inspection -

Requirements that are well defined, such as the display of a picture or words, that can be tested through observing the device in question. Inspection will be utilized for the graphical user interface, user controls, and physical characteristics of the device.

Software Unit Test -
A software test will be used to simulate the execution of the code and determine its ability to perform, given an input that a function may receive. This method will be used to determine if the software is able to register measurements and furthermore determine it to be passing or failing within the constraints of a given function.

Hardware Measurements -
Each specification that defines a quantitative limit, will be measured using tools such as a DMM, Oscilloscope, or other measuring device manufactured to the precision of our specifications. This equipment will be used alongside laboratory power supplies and function generators in order to simulate the scenario being tested. This will be used to determine the ability for ADC's, DAC's, and the current sensors to provide correct data within their given circuits.

Integration Test -
In the cases where an input voltage or current is to be measured by the device, a function generator will be used alongside a unit test. This test will determine whether the software can measure the hardware input within the accuracy specified.

Appendix O - Alternative Subject Board Integration

A top-down approach to repurposing the MicroDev code base and hardware for an unsupported subject board. WARNING: Adding or altering code may affect functionality of the MicroDev device as a whole.

New Development Board Footprint

The MicroDev system can be adapted for a development board with less than 62 pins to be under test. In order to connect a new board, it must be made to connect through a usb port for serial communication and an adapter must be created to connect the Littlefoot to the development board footprint.

A simple way to do that would be to use an Arduino Prototyping Board as seen below, and soldering on headers to connect to the alternative development boards design:

Figure 1: Arduino Prototyping Board

Or, a generic prototyping board to fit over the STM connection to the Littlefoot PCB on the MicroDev device.

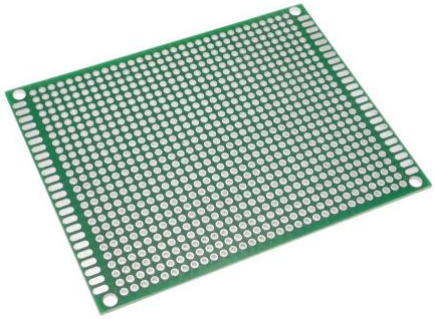


Figure 2: Generic PCB

3.3 Volt and 5 Volt Output Pin ID's

The 3.3 Volt and 5 Volt output pins are configured to a specific multiplexer address. That means that the new footprint should connect the 3V3 pin and 5V pin of the new development board to the same place as the STM Nucleo and Arduino Uno. This could also be altered in the software by changing the functions that are contained in the Model.py module.

Command Line Interface Alterations

If the board is an alternate STM Nucleo with the same footprint then only step the Board Identification and upload must be added in connection to the same configuration files of the other STM Nucleos. All code for the command line interface is within the Model.py module.

Board Identification and Upload

A new subject board will not be properly identified unless a command line interface compatible with the new development board is installed and configured. This command line interface must be able to identify the subject board through a unique ID, compile the embedded code, and also flash to the development board.

The identification of the connected development board is checked in a loop calling the board_list function. This function is in the Model.py module. The command line interface instruction receives an array of strings back from the subprocess call in Python. Below in Code Snippet 1, is an example of how the information is parsed to identify an Arduino Uno is connected.

```
659  # ------------------------------------------------------------------------
660  # Description: Uses CLI to read for subject board connections
661  # Accepts: NA
662  # Returns: [string] board type
663  # ------------------------------------------------------------------------
664  def board_list():
665
666      # 1) ARDUINO DETECTION
667      res_ardiuno = subprocess.getstatusoutput(f'arduino-cli board list')
668
669      # PARSE DATA FOR ARDUINO
670      ARD = res_ardiuno[1].split("\n")
671      boards = len(ARD)
672
673      # No Board Detection
674      if boards < 2:
675          return "No Boards Detected"
676
677      elif boards == 2:
678          print("Internet Disconnected")
679
680      # Arduino Uno Detection
681      elif boards == 3 or boards == 4:
682          board_type = ARD[1].split(" ")
683          # print(board_type[4])
684          if board_type[4] == "Serial":
685              # print(board_type[8])
686              if board_type[8] == "Uno":
687                  return "Arduino Uno Detected"
688          elif board_type[4] == "Unknown":
689              print(board_type)
690
691      else:
692          return "Overflow"
693
694
```

Code Snippet 1

The conditions for identifying the new subject board should be added after the other boards' conditionals. If nothing else is connected it will pass through those conditions and reach the connected subject board's CLI call. That should be inserted in the lines shown below in Code Snippet 2. The string that is returned after a successful detection has a place in the Controller module that is described in the pin mapping section.

```
695     # 2) STM DETECTION
696     res_stm = subprocess.getstatusoutput(f'st-info --probe')
697     # PARSE DATA FOR STM'S
698     # print(res_stm[1])
699     if len(res_stm) > 1:
700         STM = res_stm[1].split("\n")
701         boards1 = len(STM)
702     else:
703         boards1 = 0
704 
705     # STM UNIQUE IDs
706     # 0x0433: STM32F401xD/E
707     # 0x0431: STM32F411xC/E
708     # 0x0421: STM32F446
709     if boards1 > 6:
710 
711         Chip_ID = STM[5].split(" ")
712         Chip_ID = list(filter(None, Chip_ID))
713 
714         Family = STM[6].split(" ")
715         Family = list(filter(None, Family))
716         Family = str(Family[1]).replace("x", "")
717 
718         print(Chip_ID)
719         print(Family)
720         if Chip_ID[1] == "0x0431":
721             return "STM32F411 Detected"
722 
723         elif Chip_ID[1] == "0x0433":
724             return "STM32F401 Detected"
725 
726         elif Family == "F446" and Chip_ID[1] == "0x0421":
727             return "STM32F446 Detected"
728 
729     #  3) ADD NEW SUBJECT BOARD HERE
730     """
731     The Process for adding a new subject board in software...
732 
733     """
734 
735     # Use class globals for board file path and id info
736     return "No Boards Detected"
```

Code Snippet 2

Pin Mapping and Configurations

Understanding the Pin Map

The pin map connects a pin identification number, that is used in serial communication, to the multiplexer enable signal and address signals that relate to that pin. The littlefoot PCB uses those enable signals and address signals to isolate a pin to be tested. If this pin mapping is incorrect it can be difficult to debug. The enable signals control the enable pins on each of the eight multiplexers. Only one is enabled at a time. The address signals are a 0 through 7 binary address sent via output signals to control the enabled multiplexer. The disabled multiplexer will not react to the signals even though they are all on the same bus. All code dealing with the configurations is within the Controller.py module.

Use the STM32 Nucleo Pin Map in Figure 3 as a guide to writing a new pin map.

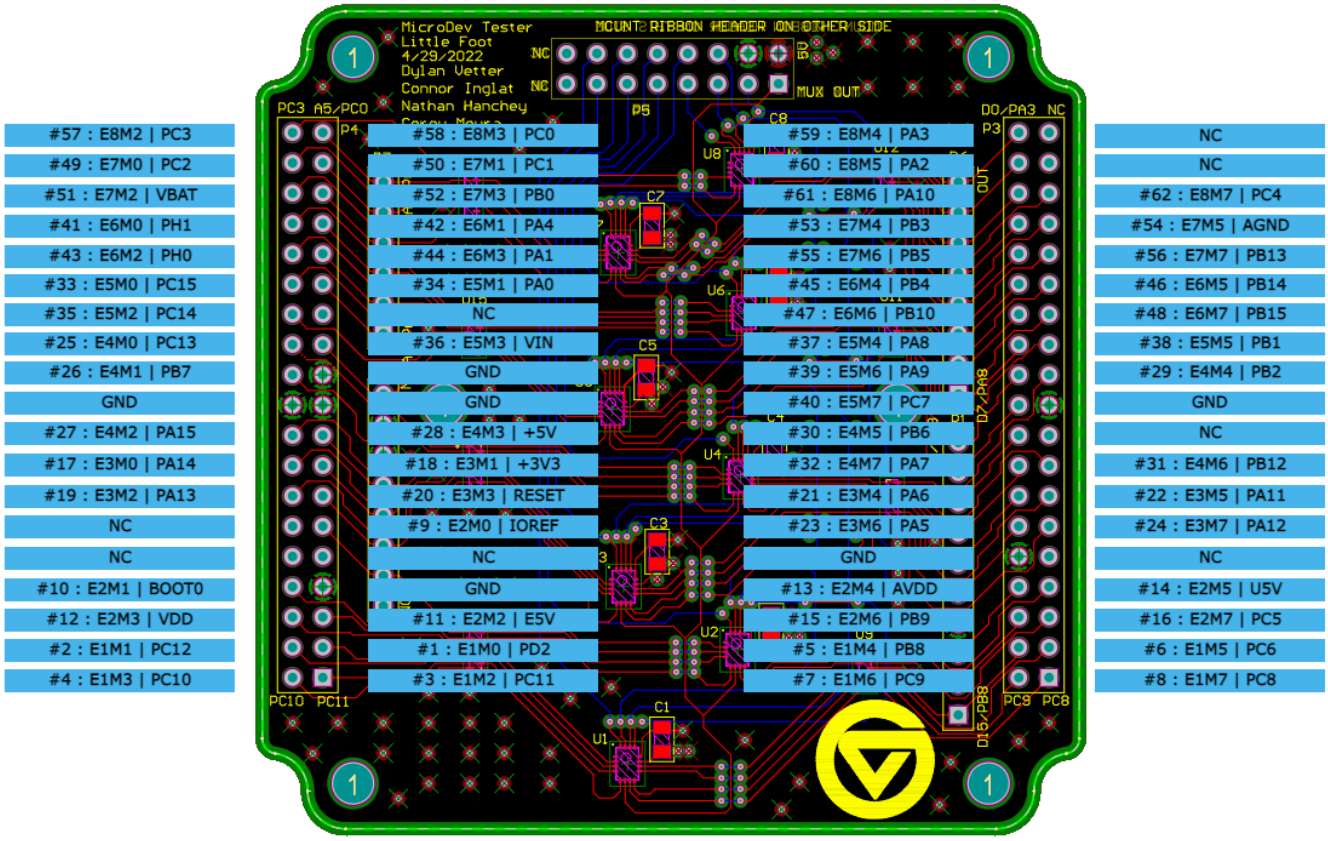


Figure 3: STM Nucleo Pin Mapping

The Pin IDs must be consistent between the test configuration file and the embedded C code receiving the pin ID value.

```
"""
Arduino Uno Mapping Key
"""
arduino_pinmap = {1: 'None', 2: 'None', 3: 'None', 4: 'None', 5: 'SCL',
                  6: 'None', 7: 'None', 8: 'None', 9: 'IOREF', 10: 'None', 11: 'None',
                  12: 'VDD', 13: 'AVDD', 14: 'None', 15: 'SDA', 16: 'None', 17: 'None',
                  18: '3V3', 19: 'None', 20: 'RESET', 21: 'D12', 22: 'None',
                  23: 'D13', 24: 'None', 25: 'None', 26: 'None', 27: 'None', 28: '5V',
                  29: 'None', 30: 'D10', 31: 'None', 32: 'D11', 33: 'None', 34: 'A0',
                  35: 'None', 36: 'VIN', 37: 'D7', 38: 'None', 39: 'D8', 40: 'D9',
                  41: 'None', 42: 'A2', 43: 'None', 44: 'A1', 45: 'D5', 46: 'None',
                  47: 'D6', 48: 'None', 49: 'None', 50: 'A4', 51: 'VBAT',  52: 'A3',
                  53: 'D3', 54: 'None', 55: 'D4', 56: 'None', 57: 'None', 58: 'A5',
                  59: 'D0', 60: 'D1', 61: 'D2', 62: 'None'}

"""
New Subject Config.
Add a Subject Mapping Key Below
"""
configurable_board = "Name Here"
configurable_filename = "fileTest.config"
configurableThreshold_filename = "fileThreshold.config"
configurable_logic = 3.3
configurable_pinmap = {1: 'None', 2: 'None', 3: 'None', 4: 'None', 5: 'None',
               6: 'None', 7: 'None', 8: 'None', 9: 'None', 10: 'None', 11: 'None',
               12: 'None', 13: 'None', 14: 'None', 15: 'None', 16: 'None', 17: 'None',
               18: '3V3', 19: 'None', 20: 'None', 21: 'None', 22: 'None',
               23: 'None', 24: 'None', 25: 'None', 26: 'None', 27: 'None', 28: '5V',
               29: 'None', 30: 'None', 31: 'None', 32: 'None', 33: 'None', 34: 'None',
               35: 'None', 36: 'None', 37: 'None', 38: 'None', 39: 'None', 40: 'None',
               41: 'None', 42: 'None', 43: 'None', 44: 'None', 45: 'None', 46: 'None',
               47: 'None', 48: 'None', 49: 'None', 50: 'None', 51: 'None',  52: 'None',
               53: 'None', 54: 'None', 55: 'None', 56: 'None', 57: 'None', 58: 'None',
               59: 'None', 60: 'None', 61: 'None', 62: 'None'}
```

Code Snippet 3

The string *configurable_board,* seen in Code Snippet 3, **must be the same string** that is returned from the board_list function added in Model.py. The high logic level of the new subject development board is to be written as the *configurable_logic*. The string for the threshold filename, *configurableThreshold_filename*, and test filenames, *configurable_filename*, must exist in the project Python folder.

The threshold file is setup as follows:

```
1   TestID,ThresholdValue1,ThresholdValue2,...
2   1,3.5,0.5
3   2,3.5,0.5
4   3,20000,50000,4.0
5   4,20000,50000,0.5
6   5
7   6,5,3.3,1.5,1,0
8   7,6
9   8,6
10  M,4,2.5
```

Code Snippet 4

After the first line, each following line is the test thresholds that will be used as the pass or fail conditions of the test. The first line describes the threshold configuration format, an improperly formatted threshold file will end the testing process. Every line shown above must be in the threshold file just as shown in Code Snippet 4. The second line represents Test #1 and each following line must be the next consecutive Test ID. M, or miscellaneous, goes on line 10. The thresholds values are as follows:

Test #1 | Min High Voltage | Max Low Voltage
Test #2 | Min High Voltage | Max Low Voltage
Test #3 | Min Resistance | Max Resistance | Pull-Up Min Voltage
Test #4 | Min Resistance | Max Resistance | Pull_Down Max Voltage
Test #5 | N/A
Test #6 | Voltages Generated By DAC (works on a loop based on length of array)
Test #7 | Current Drop mA (initial - final)
Test #8 | Current Drop mA (final - initial)
Misc. | 5V Pin Minimum Voltage | 3V3 Pin Minimum Voltage

The test file is setup as follows:

```
1   TestID,PinID,MuxAddress,MuxEnable
2   1,21,4,3
3   1,23,6,3
4   1,30,5,4
5   1,32,7,4
6   1,37,4,5
7   1,39,6,5
8   1,40,7,5
9   1,45,4,6
10  1,47,6,6
11  1,53,4,7
12  1,55,6,7
```

Code Snippet 5

After the first line, each following line is a test that will be conducted. The first line describes the test configuration format, shown in Code Snippet 5. An improperly formatted test file will end the test before finishing. The Test ID and Pin ID that is recognized by the subject development board goes first. Then, the Multiplexer Address and Enable pin comes next, all separated by commas. It is important to remember the pin ID must go with the *address* and *enable value* for that pin. The exception is with tests #7 and #8. Test #7 does not need the address and enable value and Test #8 has a preconfigured wakeup pin depending on the subject board.

Subject Board Embedded Code

Code Model for Serial Communication Data Packets

There are a few functions that must be altered in order to use the MicroDev code base for a new development board. The *main* must also be altered for initializations unique to the development board and for receiving serial data. The additional functions that must be replaced or written are as follows:

Main.c

```
int command_read(unsigned char data[]);
int command_write(unsigned int pin, unsigned int result, unsigned int test);
int analogRead(int pin);
int digitalRead(int pin);
void analogWrite(int pin, int value);
void digitalWrite(int pin, int logic);
void pinMode(int pin, int mode);
void configure_sleep_mode(unsigned int sleepmode, unsigned int interruptPin);
void wakeUp();
```

Setup.c

```
struct pin pin_set(uint32_t pin, uint32_t clock, GPIO_TypeDef * gpio,  uint8_t pin_id);
void init_pins(struct pin pins[]);
```

Within the Development folder of the MicroDev directory, there are C files to get a jumpstart with developing the new embedded code for the MicroDev system.

The following model demonstrates how to wait for serial communication on the subject board. Additionally, a method of establishing a pin mapping in the embedded code is referenced. Below is a pseudocode example of the STM32 code used to wait for serial communication for the next test.

```
INITIALIZE pin map
MAIN LOOP

IF serial message received
    Read message
    IF message passes decoding
```

```
    IF normal test
        Run Test
        Write serial message with results
    ELSE IF facade test
        Write serial message with pin register value
```

Pin Configurations and Setup

Use default configurations, auto-generated code, or the hardware abstraction layer to assist with configuring pins or power modes, if applicable.

The pins must also be identified on the subject development board to run each test. One way of doing that is to create a structure in the embedded code that has the important register values and use the index as the pin ID value, as seen in Code Snippet 7.

```
struct pin pin_set(uint32_t pin, uint32_t clock, GPIO_TypeDef * gpio,  uint8_t pin_id) {

    struct pin P;
    P.pin = pin;
    P.pin_id = pin_id;

    P.clock = clock;
    P.GPIO = gpio;

    return P;
}

// Initialize pin struct array
void init_pins(struct pin pins[]) {

    // Pin 0 Example | PA5
    pins[0] = pin_set(0x05, 0x01, GPIOA, 0);

    // Pin 1 | PD2
    pins[1] = pin_set(0x02, 0x08, GPIOD, 1);

    // Pin 2 | PC12
    pins[2] = pin_set(0x0C, 0x04, GPIOC, 2);

    // Pin 3 | PC11
    pins[3] = pin_set(0x0B, 0x04, GPIOC, 3);

    ...
}
```

Code Snippet 7

Test Inputs and Outputs

The following figure, Figure 4, describes what is generally communicated through serial communication.

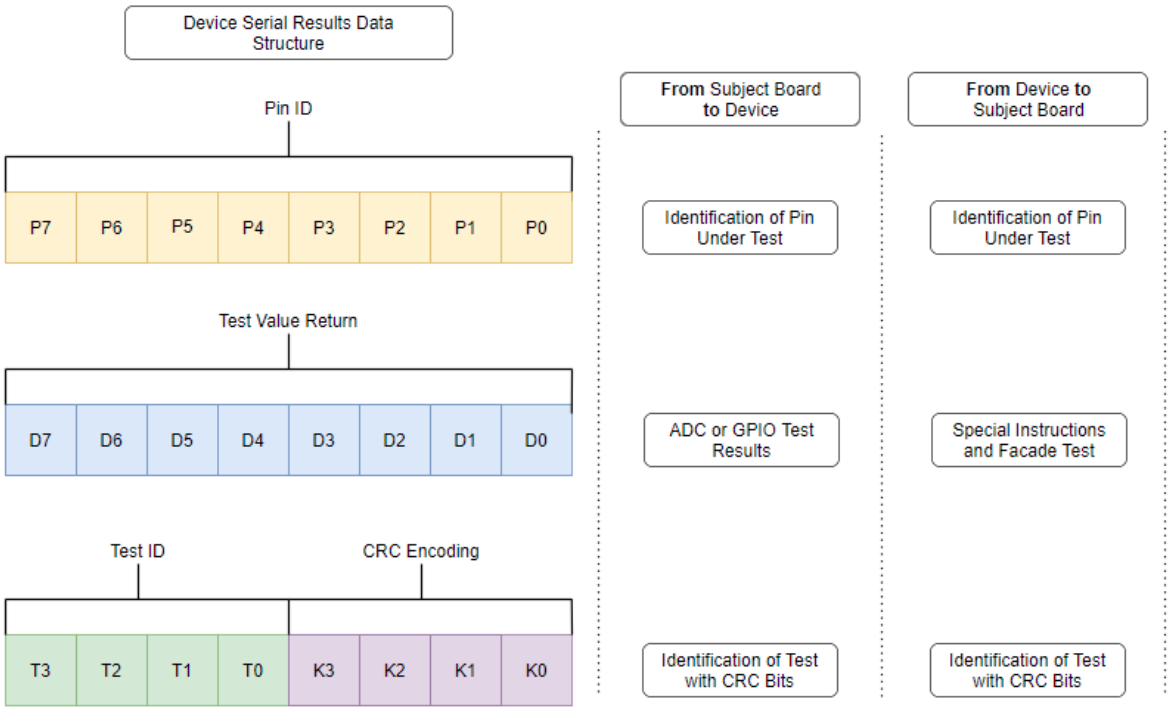


Figure 4: Serial Communication Data Packets

Specifically, each test has its own Test ID value and some of them return a measurement in an 8-bit number. A description of this is below.

Output Test #1 - Sets Voltage with no returned measurement

Output Test Under Load #2 - Sets Voltage with no returned measurement

Input Pull Up Test #3 - Sets Voltage with no returned measurement

Input Pull Down Test #4 - Sets Voltage with no returned measurement

Input Logic Test #5 - Sets Input and Returns a 1 or 0 for High or Low digital signal.

Analog to Digital Converter Test #6 - Sets ADC input and return value shifted to 8 bits.

Power Mode Test #7 - Sets Power Mode and returns nothing

Wakeup Test #8 - Wakes up for low power mode

Testing and Verifying New Code

Refer back to software verification in the design document.

Appendix P - Hardware Validation Testing Results

HARDWARE VALIDATION

	Test not required
	Hardware Test
	Software / Hardware Test
	Software Test

Spec #	Validation Method	NOTES	Board 1	Board 2	Board 3	Board 4	Board 5	Board 6	Board 7	Board 8	Board 9	Board 10
TVS DIODE TESTS												
2.02.1	Transient voltage suppression (TVS) diodes are used for the USB connector where the user connects their subject board to. The overvoltage protection of this IC is well documented in its datasheet.	Inspection. Over voltage testing and destructive testing is not needed										
ADC TESTS												
2.05.1	The ADC must measure voltage with an accuracy of 0.1V. Using a separate power supply, 5V will be applied to the ADCs input channel. Using a DMM, the voltage at the ADC's output terminal will be measured. This measured value will be compared against the ADC's measured value to be within 0.1V of the DMM's value.	**DMM Measurement:** TP4 to GND	DMM / ADC Measurements: 0.0V: 0.00 / 0.00 1.0V: 0.99 / 0.99 3.3V: 3.28 / 3.28 5.0V: 4.97 / 4.97	DMM / ADC Measurements: 0.0V: 0.0005 / 0.0 1.0V: 1.003 / 0.991 3.3V: 3.303 / 3.283 5.0V: 5.003 / 4.976	DMM / ADC Measurements: 0.01V: 0.012 / 0.010 1.0V: 1.002 / 1.001 3.3V: 3.302 / 3.314 5.0V: 5.003 / 4.992	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.986 3.3V: 3.29 / 3.28 5.0V: 4.99 / 4.98	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.991 3.3V: 3.29 / 3.27 5.0V: 4.99 / 4.98	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.996 / 0.986 3.3V: 3.29 / 3.28 5.0V: 4.99 / 4.98	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.986 3.3V: 3.29 / 3.31 5.0V: 4.99 / 5.01	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.991 3.3V: 3.29 / 3.29 5.0V: 4.97 / 4.96	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.991 3.3V: 3.29 / 3.28 5.0V: 4.99 / 4.98	DMM / ADC Measurements: 0.0V: 0.0 / 0.0 1.0V: 0.995 / 0.991 3.3V: 3.29 / 3.28 5.0V: 4.99 / 4.98
			Board 9	**Board 10**								
3.02.1	Through USB serial communication, the Raspberry Pi will program the arduino to configure a pin as a GPIO output. This output voltage will be measured using a DMM and will be compared against the ADC's voltage measurement.	**DMM Measurement:** Configured Pin to GND Testing we can put configure SB pin to GPIO output and measure its voltage	DMM : 5.09 ADC : 5.11	DMM : 5.09 ADC : 5.11								
3.03.1	Through USB serial communication, the Raspberry Pi will program the arduino to configure a pin as a **GPIO input pullup**. The GPIO pin will be connected to the onboard load resistor (~390 Ohm). The voltage at the ADC will be measured using a DMM. The resistance of the (~390 ohm) load will also be measured using a DMM. The measured voltage and resistance will be used to calculate the internal resistance of the pin. This value will be compared against the device's calculation of the internal resistance.	**DMM Measurement:** TP4 to GND Expected Range: 0.039667V - TO - 0.097948V **Rcalc = ((5 * 390) / Vmeasured) - 390** Pull-Up Resistor(20K-50K) Load (R8) = 390 Ohms	DMM Output Voltage: 0.0553 DMM Load Resistance: 390 ohm ADC Voltage: 0.051 DMM Rcalc: 34.872 kohm Device Rcalc: 37.845 kohm	DMM Output Voltage: 0.0556 DMM Load Resistance: 388 ohm ADC Voltage: 0.050 DMM Rcalc: 34.504 kohm Device Rcalc: 38.412 kohm								

3.03.1 Extension	Through USB serial communication, the Raspberry Pi will program the arduino to configure a pin as a **GPIO input pulldown**. The DAC will generate a voltage for the GPIO pin. The voltage at the ADC input will be measured using a DMM. The measured voltage will be used to calculate the internal resistance of the pin. This value will be compared against the device's calculation of the internal resistance.	**DMM Measurement:** TP4 to GND (Arduino does not have internal pulldown) **Rcalc = (2000*Vmeasured) / (3.3V - Vmeasured)** Special Note: Pin 13 (LED Connected) ADC - 2.13V (3.6K internal)	DMM Output Voltage: 3.12V DMM Load Resistance: 1.994 kohm ADC Voltage: 3.09V DMM Rcalc: 34.562 kohm Device Rcalc: 29.340 kohm	DMM : 3.12V DMM Rload: 1.995k ohm ADC : 3.09V DMM Calc. R: 34.580 kohm Device Calc. R: 29.335 kohm								
3.04.1	Through USB serial communication, the Raspberry Pi will program the arduino to configure a pin as a GPIO **input pull-up**. The voltage of the pin will be measured using a DMM. This voltage measurement will be compared against the ADC's voltage measurement. (The ADC's voltage measurement must be within 0.1V of the DMM's voltage measurement.)	**DMM Measurement:** Pin to GND	DMM : 4.91 ADC : 4.73	DMM : 4.90 ADC : 4.73								
3.04.1 Extension	Through USB serial communication, the Raspberry Pi will program the arduino (FIX: STM not Arduino) to configure a pin as a GPIO **input pull-down**. The voltage of the pin will be measured using a DMM. This voltage measurement will be compared against the ADC's voltage measurement. The ADC's voltage measurement must be within 0.1V of the DMM's voltage measurement.	**STM ONLY** DMM Measurement: Pin to GND	DMM : 3.11 ADC : 3.09	DMM : 3.12 ADC : 3.09								
INA TESTS												
2.06.1	The current sensor must measure current with an accuracy of 10µA up to 1mA. Using a separate power supply, a current of 0.5mA will be supplied through the shunt resistor via the test points located on the PCB. This current will be measured using a DMM and will be compared against the IC's measured value to be within 10µA.	INA219 May be allowed to delete this test since subject board current does not dip below 1mA ever, even in sleep mode										
2.06.2	The current sensor must measure current with an accuracy of 10mA from 0-1A. Loads of 5, 10, 20, 50, and 500 Ω will be connected through the USB terminal. Current out of this terminal will be measured with a DMM and compared against BigFoot's measurement. BigFoot's measured value must be within 10mA.	INA219 Removed filter from boards: 2 / 6 / 10	DMM / INA (DMM - INA) 5: 840.1 / 839.4 (0.7) 10: 479.1 / 479.6 (0.5) 20: 252.1 / 252.4 (0.3) 50: 103.2 / 102.9 (0.3) 500: 9.0 / 8.80 (0.2)	DMM / INA (DMM - INA) 5: 849.0 / 848.6 (0.4) 10: 481.3 / 484.5 (3.2) 20: 252.7 / 256.7 (4.0) 50: 103.5 / 106.8 (3.5) 500: 9.0 / 13.7 (4.7)	DMM / INA (DMM - INA) 5: 838.0 / 845.9 (7.9) 10: 476.4 / 482.5 (6.1) 20: 250.8 / 255.1 (4.3) 50: 103.0 / 106.3 (3.3) 500: 9.0 / 11.70 (2.7)	DMM / INA (DMM - INA) 5: 838.4 / 842.5 (4.1) 10: 477.5 / 483.9 (6.4) 20: 252.0 / 254.1 (2.1) 50: 102.9 / 104.9 (2.0) 500: 9.1 / 10.10 (1.0)	DMM / INA (DMM - INA) 5: 851.5 / 854.4 (2.9) 10: 480.9 / 484.1 (3.2) 20: 252.8 / 257.5 (4.7) 50: 103.2 / 108.0 (4.8) 500: 9.0 / 13.10 (4.1)	DMM/INA & DMM - INA 5: 782.0 / 784 (2.0) 10: 451.0 / 456.3 (5.3) 20: 238.0 / 241.0 (3.0) 50: 96.2 / 99.4 (3.2) 500: 8.9 / 11.0 (2.1)	DMM/INA & DMM - INA 5: 850.4 / 858.9 (8.5) 10: 479.9 / 484.9 (5.0) 20: 252.2 / 258.2 (6.0) 50: 103.0 / 108.0 (5.0) 500: 9.0 / 12.3 (3.3)	DMM/INA & DMM - INA 5: 847.9 / 841.0 (6.9) 10: 480.7 / 478.2 (8.5) 20: 252.6 / 253.4 (0.8) 50: 103.4 / 104.9 (1.5) 500: 9.0 / 12.3 (3.3)	DMM/INA & DMM - INA 5: 850.8 / 856.4 (6.4) 10: 480.7 / 486.6 (5.9) 20: 252.0 / 259.1 (7.1) 50: 103.5 / 109.0 (5.5) 500: 9.0 / 13.5 (4.5)	DMM/INA & DMM - INA 5: 794 / 802.9 (8.9) 10: 456.5 / 463.0 (6.3) 20: 246.8 / 250.2 (3.4) 50: 102.6 / 104.5 (1.9) 500: 9.1 / 9.99 (0.89)
3.09.1	The device will configure a subject board for sleep mode. A DMM will be used to measure the voltage drop across the shunt resistor. A DMM will be used to measure the resistance of the shunt resistor. The voltage and resistance measurements will be used to calculate the current through the shunt resistor. This calculation will be compared against the current measurement IC's measurement.	Satisfied by 2.06.2 and 3.08.1 / 3.08.2										
DAC TESTS												
3.06.1	The device will configure the DAC to output several voltages within the range of logic low and logic high of the subject board. A DMM will be used to measure	DMM Measurement: TP2 (Vout) to TP7 (GND)	DMM Measurements: 0.0V: 0.004 1.0V: 0.99 2.25V: 2.23	DMM Measurements: 0.0V: 0.003 1.0V: 1.0 2.25V: 2.24	DMM Measurements: 0.0V: 0.001 1.0V: 1.004 2.25V: 2.2515	DMM Measurements: 0.0V: 0.001 1.0V: 0.990 2.25V: 2.240	DMM Measurements: 0.0V: 0.002 1.0V: 1.0 2.25V: 2.249	DMM Measurements: 0.0V: 0.0025 1.0V: 1.00 2.25V: 2.24	DMM Measurements: 0.0V: 0.0041 1.0V: 1.00 2.25V: 2.24	DMM Measurements: 0.0V: 0.0023 1.0V: 0.99 2.25V: 2.24	DMM Measurements: 0.0V: 0.0024 1.0V: 1.00 2.25V: 2.24	DMM Measurements: 0.0V: 0.0012 1.0V: 0.992 2.25V: 2.23

	the output voltage of the DAC. The measured voltage will be compared against the voltage code the DAC was provided.	Provided Voltages To DAC	3.3V: 3.27 5.0V: 4.96	3.3V: 3.29 5.0V: 4.98	3.3V: 3.297 5.0V: 5.011	3.3V: 3.280 5.0V: 4.970	3.3V: 3.29 5.0V: 4.99	3.3V: 3.29 5.0V: 4.98	3.3V: 3.29 5.0V: 4.98	3.3V: 3.29 5.0V: 4.99	3.3V: 3.29 5.0V: 4.99	3.3V: 3.28 5.0V: 4.98
3.07.1	The device will configure the DAC to output several voltages within a range for testing the subject board's ADC. A DMM will be used to measure the MUX out voltage. The measured voltage will be compared against the mux voltage code the DAC was provided. (Test ensures analog switches are working correctly)	DMM Measurement: TP(TP7) to (On MUXout pin) Ask Baine about the MUX out Voltage	DMM Measurements: 0.0V: 0.004 1.0V: 0.99 2.25V: 2.24 3.3V: 3.29 5.0V: 4.98	DMM Measurements: 0.0V: 0.000 1.0V: 0.999 2.25V: 2.24 3.3V: 3.28 5.0V: 4.97	DMM Measurements: 0.0V: 0.01 1.0V: 1.00 2.25V: 2.25 3.3V: 3.29 5.0V: 5.00	DMM Measurements: 0.0V: 0.001 1.0V: 0.997 2.25V: 2.241 3.3V: 3.280 5.0V: 4.970	DMM Measurements: 0.0V: 0.002 1.0V: 1.0 2.25V: 2.248 3.3V: 3.29 5.0V: 4.98	DMM Measurements: 0.0V: 0.0025 1.0V: 1.00 2.25V: 2.24 3.3V: 3.29 5.0V: 4.97	DMM Measurements: 0.0V: 0.0042 1.0V: 1.00 2.25V: 2.24 3.3V: 3.29 5.0V: 4.97	DMM Measurements: 0.0V: 0.004 1.0V: 0.999 2.25V: 2.24 3.3V: 3.29 5.0V: 4.98	DMM Measurements: 0.0V: 0.0024 1.0V: 1.00 2.25V: 2.24 3.3V: 3.29 5.0V: 4.98	DMM Measurements: 0.0V: 0.0013 1.0V: 0.991 2.25V: 2.23 3.3V: 3.28 5.0V: 4.96
MUX TESTS												
LittleFoot	LittleFoot will be controlled using a separate power supply. Enable and Address pins will be supplied 5V in combinations that will test each MUX pin at least once. When a pin is enabled, it will be supplied 3.3V. This voltage will be checked using a DMM on the MUX output signal to be close to the 3.3V provided.	board 1 is delicate	Passed	Passed	Passed	Passed	Passed	Passed	Passed	Passed	Passed	Passed
ANALOG SWITCH TESTS												
	In the case that the ADC or DAC is responding to I2C but test results indicate that the subject board is not connected, it is likely that the one or more of the analog switches are not connected properly	If this is occuring: 1. Examine the connections on the analog switches for bridging/shorts 2. Replace IC if still no connection after examining										
LEVEL SHIFTING TESTS												
	If no IC communication is established from the raspberry pi to the INA / ADC / and DAC on the BigFoot PCB, then its likely that the LSF0102DCTR level shift ICs is faulty.	If this is occuring: 1. Check for bridging on the level shift IC pins 2. Replace IC if still no connection after examining										
SLEEP MODE TESTING												
			Board 9	**Board 10**								
3.08.1 / 3.08.2	A subject board will be placed in sleep mode.The DAC output voltage will then be routed to the subject board's wakeup pin. Current draw of the subject board will be monitored before and after to confirm that the device was put to sleep and awoken from its sleep mode.	Arduino: **"Idle Mode"**	**INA Measurements:** Normal Current: 0.0745 Sleep Current: 0.06375 Normal Current: 0.07375 **Difference: ~10.75mA**	**INA Measurements:** Normal Current: 0.0745 Sleep Current: 0.064 Idle Current: 0.755 **Difference: ~10.5mA**								
		Arduino: **"ADC Noise Reduction**	**INA Measurements:** Normal Current: 0.0755 Sleep Current: 0.06175 Normal Current: 0.07475 **Difference: ~13.75mA**	**INA Measurements:** Normal Current: 0.07425 Sleep Current: 0.0615 Normal Current: 0.0745 **Difference: ~12.75mA**								

Arduino: "Power Down Mode"	DMM Measurements: Normal Current: 0.07575 Sleep Current: 0.0515 Normal Current: 0.07475 **Difference: ~24.25mA**	DMM Measurements: Normal Current: 0.0745 Sleep Current: 0.0525 Normal Current: 0.0745 **Difference: ~22.0mA**
Arduino: "Power Save Mode"	INA Measurements: Normal Current: 0.07525 Sleep Current: 0.05475 Normal Current: 0.07525 **Difference: ~20.05mA**	INA Measurements: Normal Current: 0.07425 Sleep Current: 0.054 Normal Current: 0.074 **Difference: ~20.25mA**
Arduino: "Standby Mode"	INA Measurements: Normal Current: 0.0725 Sleep Current: 0.05225 Normal Current: 0.0755 **Difference: ~20.25mA**	INA Measurements: Idle Current: 0.0735 Sleep Current: 0.0515 Idle Current: 0.0735 **Difference: ~22.0mA**
Arduino: "Extended Standby Mode"	INA Measurements: Normal Current: 0.0755 Sleep Current: 0.0550 Normal Current: 0.0740 **Difference: ~20.05mA**	INA Measurements: Normal Current: 0.07525 Sleep Current: 0.5425 Normal Current: 0.07425 **Difference: ~21.0mA**
STM: "Sleep Mode" main power regulator on	INA Measurements: Normal Current: 0.08925 Sleep Current: 0.0745 Normal Current: 0.0890 **Difference: ~14.75mA**	INA Measurements: Normal Current: 0.08825 Sleep Current: 0.0745 Normal Current: 0.0880 **Difference: ~13.75mA**
STM: "Stop Mode" low power regulator on	INA Measurements: Normal Current: 0.0880 Sleep Current: 0.0655 Normal Current: 0.0885 **Difference: ~23.0mA**	INA Measurements: Normal Current: 0.0880 Sleep Current: 0.06675 Normal Current: 0.08825 **Difference: ~21.25mA**

		STM: **"Standby Mode"**	**INA Measurements:** Normal Current: 0.08875 Sleep Current: 0.0630 Normal Current: 0.08875 **Difference: ~25.75mA**	**INA Measurements:** Normal Current: 0.0885 Sleep Current: 0.0630 Normal Current: 0.08875 **Difference: ~25.5mA**										
INTEGRATION TESTING														
	Using an STM board without any faults, BigFoots and LittleFoots are placed in pairs used to test the working STM board. The test is expected to pass in all categories, proving that the BigFoot and LittleFoot Pair are working properly.	BigFoots												
	Using an STM board without any faults, BigFoots and LittleFoots are placed in pairs used to test the working STM board. The test is expected to pass in all categories, proving that the BigFoot and LittleFoot Pair are working properly.	LittleFoots												