Development of a Portable, Automated, MRI-Compatible Tactile Stimulation Device

By

Nicole Baptist

M.S. Northwestern University, 2021

Project Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Robotics
in the
Department of Mechanical Engineering

Executive Summary

This research project contributed to the Robotics and Sensorimotor Control Lab's efforts in improving our understanding of human somatosensory perception in their upper limbs. The overall project goal is to identify the reason for tactile perception deficits in individuals with stroke. My contribution to this project was to advance a proof-of-concept mechatronic device to one that could be used for pilot testing. My contributions to this project include developing the hardware and software to permit automated control of the mechanical stimuli applied to an individual's fingertip. The system now includes, in addition to the actuator fabricated in the previous quarter, a pressure regulator, pressure sensor, PIC32-powered controller, and serial communication via a computer for automated control of and data collection from the device.

Table of Contents

Introduction

The purpose of this device is to control and quantify a tactile stimulus at an individual's fingertip in a magnetic resonance imaging (MRI) environment.

1.1 Scope

The scope of the project is to apply a tactile stimulus to an individual's fingertip in an MRI environment. Requirements for the device include sufficient robustness and flexibility for travel between testing environments and future development. The pneumatic device consists of six (6) main components connected via polyurethane tubing and polypropylene connectors (for pneumatic output) and cables (for electrical control). An air supply capable of outputting at least 20 psi is also required. The system pipeline, as visualized in Figure 1.1.1 below, begins with a PIC32 microcontroller. This microcontroller controls the solenoid (to designate the receiving chamber of the actuator) and the pressure regulator output. A Honeywell pressure sensor is incorporated in the path to the bottom chamber only for additional feedback on the accuracy of the pressure regulator output before the output pressure reaches the polytetrafluoroethylene (PTFE) end-effector. Accurate control of the top chamber is not feasible due to air leaks through the opening for the piston, so there is no sensor or regulator connected to the top chamber.

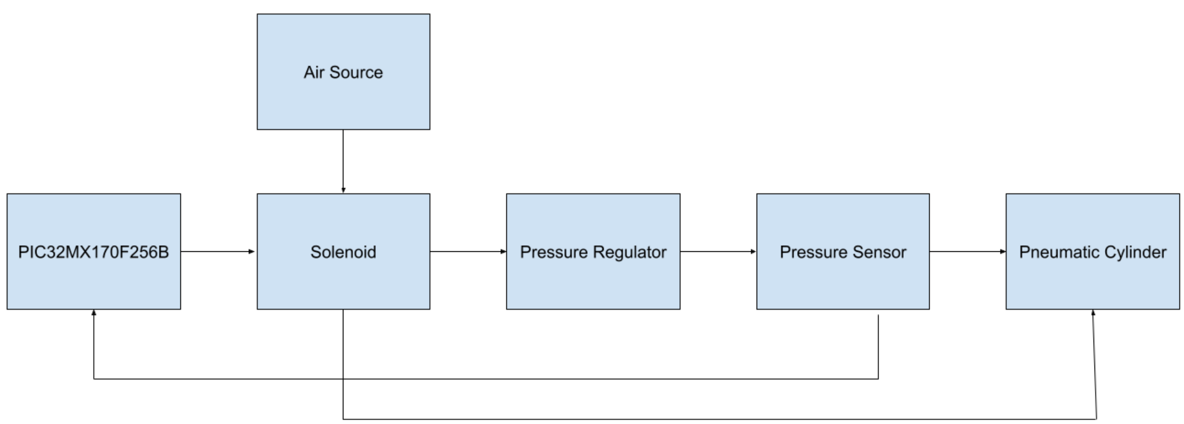


Figure 1.1.1: *Pipeline of project system*.

1.2 Requirements and Deliverables

The following requirements were placed on the project for the corresponding justifications:

MRI Compatibility - The tactile stimulation will be analyzed using functional magnetic resonance imaging (fMRI). The device and all connected components within the MRI room must be composed of non-magnetic materials. The strength of typical scanners, such as those used at the Northwestern University Center for Translational Imaging, range from 1.5T to 3T. Extreme caution is required with any conductive materials due to the potential dangers of the MRI unit, such as radio-frequency (RF) heating and the conversion of "normal" objects of magnetic material into projectiles.

Robustness - Due to the frequent travel between the Evanston and Chicago campuses required for testing, the revised design must be neatly containable with minimal assembly and disassembly required. All components should be securely soldered or utilize the proper dual in-line package (DIP/DIL) socket. Control should be executed through serial commands to limit hardware requirements and eliminate any unnecessary areas of potential failure as much as possible. Additionally, a custom case to protect the controller circuit must be designed and implemented.

Automatability - In order to streamline the testing procedure (and also to minimize the required hardware while still maintaining all controller capabilities), control of the system should be executed purely digitally through serial communication. This also ensures that all controls are clearly labeled and in one place to make control intuitive for all users, regardless of technical background.

The following deliverables are required for the completion of this project:

Documentation - Information in this report will serve as a manual for the device. Specifically, this manual will cover the design, assembly, and any notable limitations of the system. When appropriate, links to source files, computer aided design (CAD) files, datasheets, and other supporting documents will be provided and cited. Rationales will also be provided for design decisions.

Fabrication - Manufacture of the pneumatic cylinder, printed circuit board (PCB), controller case, hand interface, and all additional hardware components required must be completed and assembled so that the device is ready to use upon the project's completion.

Demonstration - A demonstration of controller features for future reference is also required. Any user, regardless of technical background, should have at least a basic understanding on how to use the device from the recorded demonstration(s).

System Design

2.1 Circuit Design

The control system was designed using the PIC32MX170F256B microcontroller, as it is popular with developers in industry and allows for significantly greater speed, memory, and additional capabilities in comparison to other possible controllers geared more towards hobbyists. It is also well-documented. The PIC receives power (after passing through an L4931 voltage regulator) and communicates with the user’s computer via a FTDI-USB cable. It is programmed using an MPLAB SNAP produced by Microchip Technology.

The pressure regulator and solenoid valve require a 24 V power source, which is provided through a power supply. This 24 V input is also converted to 5 V, using a DC-to-DC converter, to power the H-bridges and pressure sensor.

Adding on to the original design by Alexander Hay, the new circuit now features i) a new digital pressure regulator that is both more available and affordable, ii) slide switches to simplify testing, iii) a low-pass resistor-capacitor (RC) filter with a pulse-width modulation (pwm) signal to control the new regulator via analog output signal, and iv) a PCB, which includes a development area for added flexibility with new components. Note that the slide switches allow the user to quickly connect and disconnect the source power, like that of a light switch, to ensure that there is no shorting while navigating through the circuit components.

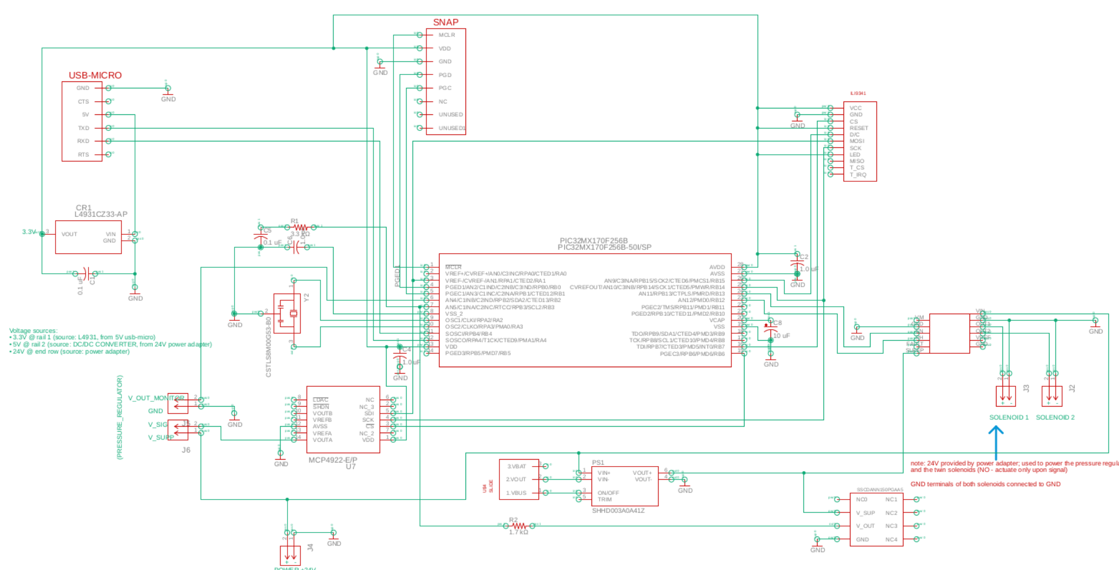


Figure 2.1.1: *Schematic of the system*.
Note: files for circuit schematics and boards are in the git repository, and a summary of components can be found in the appendix

2.2 Pneumatic Cylinder Design

The system was built using components from the MRI-Compatible Tactile Stimulation Device created by Alex Hay. The actuator was designed loosely based off of the air cylinder designs of iPolymer, in collaboration with experts at Northwestern's Segal Prototyping and Fabrication Lab.

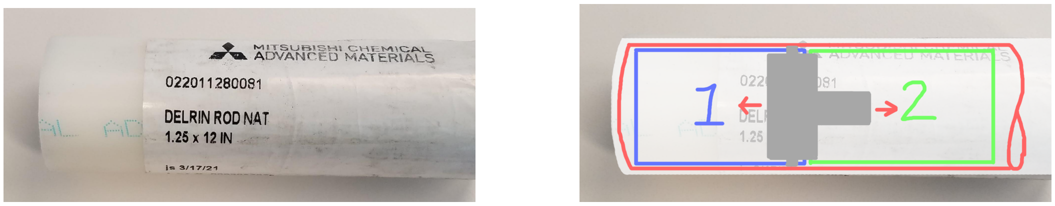


Figure 2.2.1: *Chambers of PTFE double-action cylinder*.
(Note: Although the figure depicts a Delrin rod, the Delrin model was constructed, but not used)

Numerous materials were initially considered for the mechanical design, including PTFE (Teflon), Delrin/Acetal, high-density polyethylene (HDPE), and nylon. PTFE was chosen as the cylinder material for its natural slipperiness, machinability, and desirable results in materials testing, as previously conducted by Professor Netta Gurari. The cylinder features two chambers to push the enclosed piston up and down the length of the cylinder. The barrel was machined and tapped first using a conversational mill, with calculated dimensions inspired by the iPolymer design. The piston and end cap were subsequently machined on the lathe after finding a proper fit for the outer diameter using a set of pin gages. O-ring gauge dimensions[3] were also used to determine the suitable o-ring compression. The resulting dimensions can be viewed in the part drawings included in the appendix. The holes for the Polypropylene Push-to-Connect Tube Male Connectors[4] were made at the ends of the two chambers to ensure that air would flow into the desired chamber regardless of the piston's position while also avoiding blockage by the end cap. Calculations were done to ensure that the two lines of tubing had a direct and unobstructed connection to their respective chambers, as seen in Figure 2.2.2 on the next page. In the current design, the two holes are offset by a little less than 90º to ensure that the connectors do not interfere with each other due to the cylinder's short length.

3: From Marco Rubber and Plastics' O-Ring Design Considerations article, it was determined that only approximately 20% of the o-ring should be exposed; the dimensions of the groove were calculated accordingly.

4: The SharkBite and Push-to-Connect connectors are both barbed to ensure that the connections are secure and air-tight; tubing can be easily removed by holding the barbed piece in place while pulling the tubing outward. (please see Figure 9 in the Assembly and Directions section)

Figure 2.2.2: *Final cylinder design*.

3.1 Major Components

Air Supply

The California Air Tools P1060S provides 1.20 CFM @ 90 psi, has a one gallon tank, and runs at 56dB. It can be used as a portable air source when there is no wall outlet present.

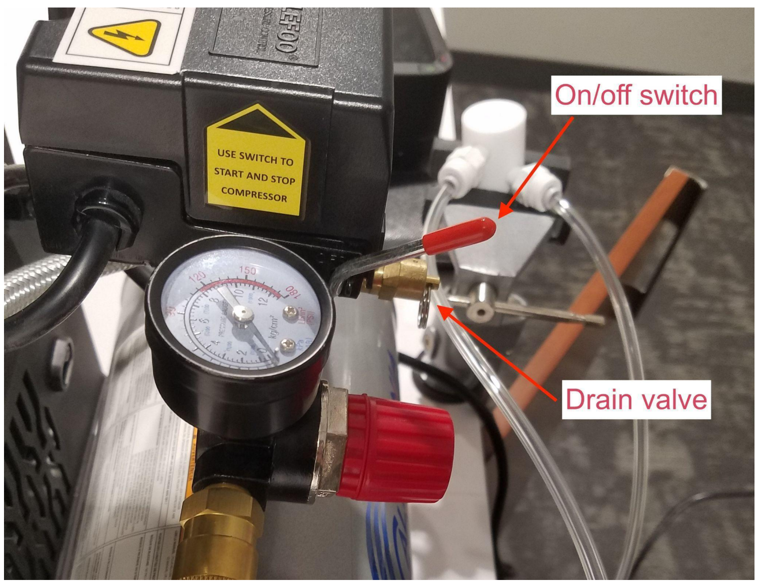


Figure 3.1.1: *Compressor controls.*

Note: the purpose of the drain valve is to release excess pressure at the end of testing, but before storage of the device, as a safety precaution

Solenoid

A 3-position solenoid from AutomationDirect is used to direct the airflow between the two chambers of the pneumatic actuator. The two solenoids that make up the solenoid valve take turns turning on or off to allow air flow from the supply to the pneumatic cylinder. Both solenoids require 24 V to actuate and are controlled through a single Pololu DRV8256E Single Brushed DC Motor Driver Carrier. Air is directed to either the pressure regulator (which is connected to the bottom chamber of the cylinder) or directly to the top chamber. A pair of Murrelektronik solenoid valve cables were purchased to connect the solenoids to the H-bridge.

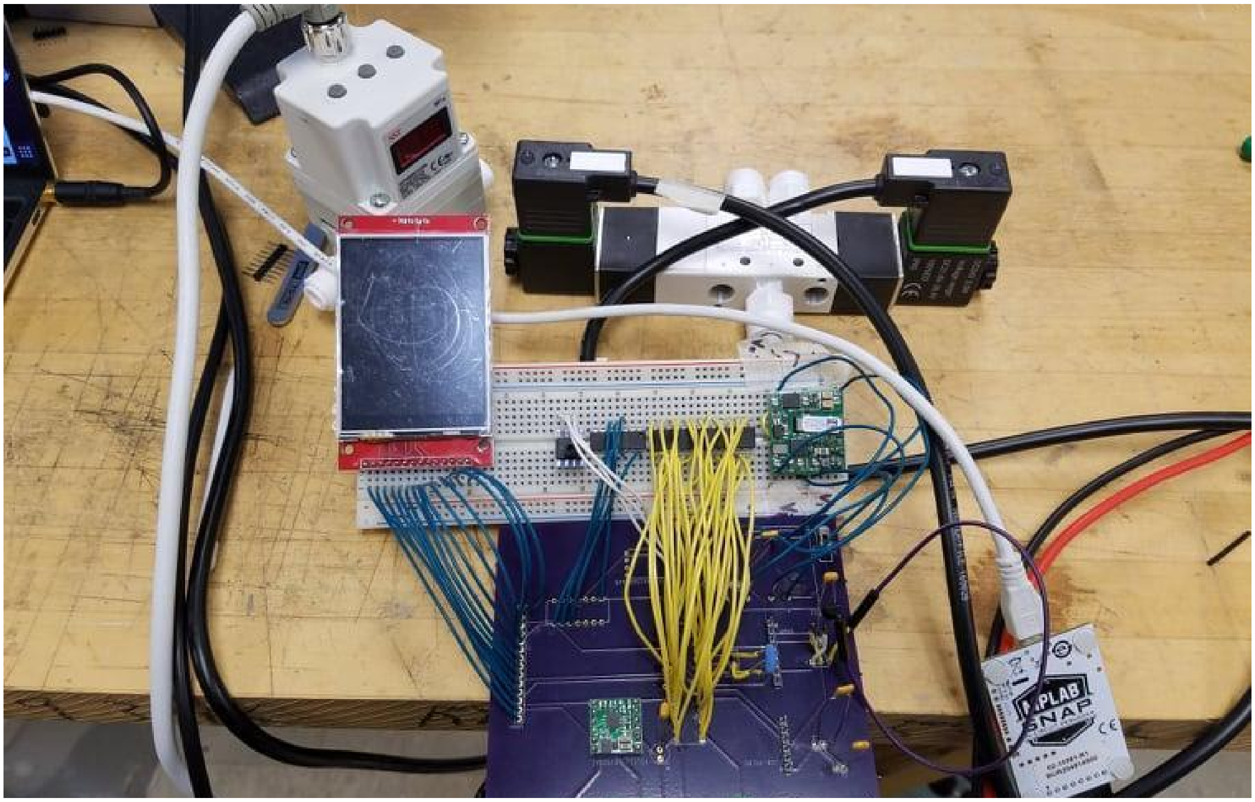


Figure 3.1.2: *Diagram of solenoid connections.*

Regulator

The NNT digital pressure regulator valve receives air from one solenoid output and an analog input signal from the PIC32 to control the pressure output to the bottom chamber of the pneumatic cylinder. Note that its absolute maximum and minimum pressure can be set beforehand to avoid reaching the pain threshold of approximately 400 kPa and ensure the safety of the test subject.

H-bridge

The aforementioned Pololu DRV8256E Single Brushed DC Motor Driver Carrier is a versatile H-bridge. One is used to control both sides (left and right) of the solenoid valve's actuation and the other controls the pressure output of the regulator. The driver receives an input voltage, an enable input in the form of a pwm signal, a phase output in the form of a general purpose input/output signal, and a sleep input between 1.8 and 5.5 V. It has two output pins.

Actuator

The dual chamber cylinder was custom made for this project. The barrel, piston, and end cap were all prepared out of polytetrafluoroethylene, otherwise known as PTFE or Teflon. These components were machined on the conversational mill and turned on the lathe. The end cap and the piston designs each incorporate one Buna-N o-ring to seal the two chambers and minimize air leaks. A ¼” tapered polypropylene (PP) connector was tapped into each barrel for separate supply lines from the two output ports of the twin solenoids. The audible hissing during piston retraction suggests that there is significant leakage from the opening for the piston at the top of the cylinder. Despite this, such leakage should not drastically impact the goal of providing accurately controlled tactile stimuli since it mainly occurs when the piston is retracted. While there is likely some leakage from the bottom chamber, as this is a hand-made device, there is no obvious auditory or visual cue to suggest significant leakage from the bottom chamber.

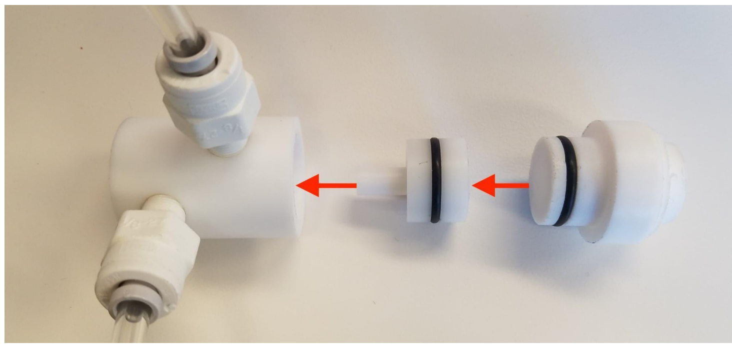


Figure 3.1.3: *Image of the pneumatic cylinder disassembled.*
From left to right: Barrel (with connections), piston, end cap.

Controller

The control system, as mentioned in Section 2.3, is controlled using a PIC32170F256B microcontroller. This model of PIC was chosen for its pin layout and familiarity from its use in the Advanced Mechatronics course. From the previous iteration of this circuit, the 6 V input and MIC2940, relay, and pressure regulator were replaced with the 24 V (already built in) source and DC-to-DC converter, Pololu H-bridges, and NNT commercial pressure regulator, respectively. The new design prioritizes portability and robustness while maintaining some flexibility. Therefore, this design eliminated the toggle switch and push buttons of the previous design in favor of pure serial communication control. The 320x240 LCD screen is kept for additional debugging purposes and as a finishing aesthetic.

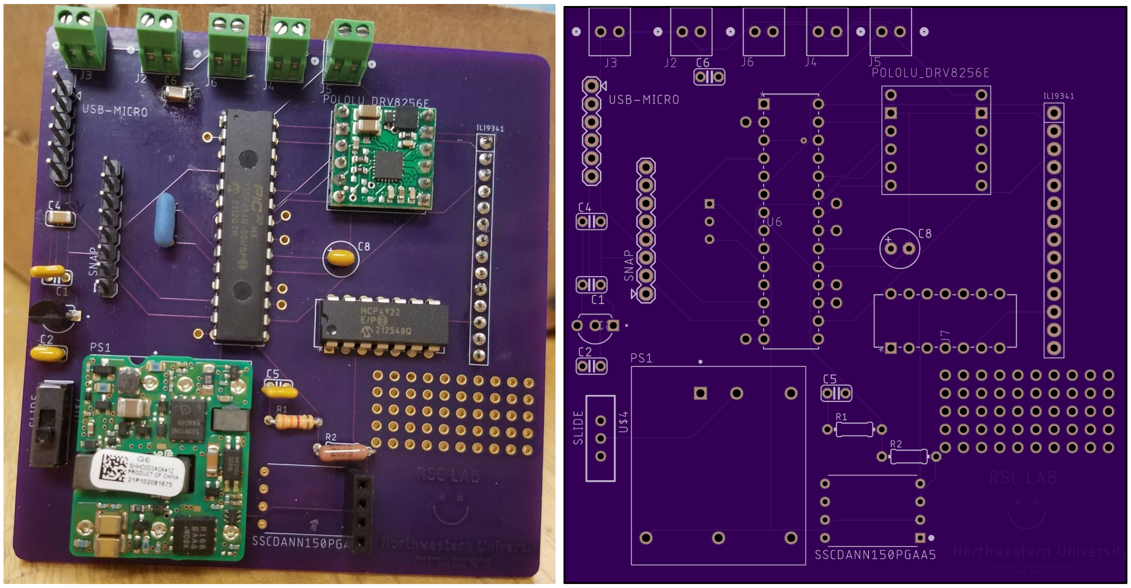


Figure 3.1.4: *Image of the controller PCB.*

3.2 Assembly

The polyurethane (PU) tubing arrangement can be seen in Figure 3.1.3 below. The power supply connects to the input of the twin solenoids while one output connects to the pressure regulator supply inlet and the other to the top chamber of the cylinder. The output of the pressure regulator connects to both the top chamber of the cylinder and the pressure sensor using a T-connector.

The back of the controller case can be opened to attach the SNAP for programming, but the SNAP should be disconnected and the controller case closed during testing to eliminate any external effects on the circuit while it is being used. Once the controller is ready for use, all that needs to be connected are the pressure regulator, two solenoids, and the 24 V power supply through the slot on the top of the controller case, the FTDI-USB cable through its own opening on the top of the case, and the tubing connecting to the Honeywell pressure sensor. The specific wire connections for the screw terminals can be found in the circuit diagram, board file, or Figure 3.2.1 below.

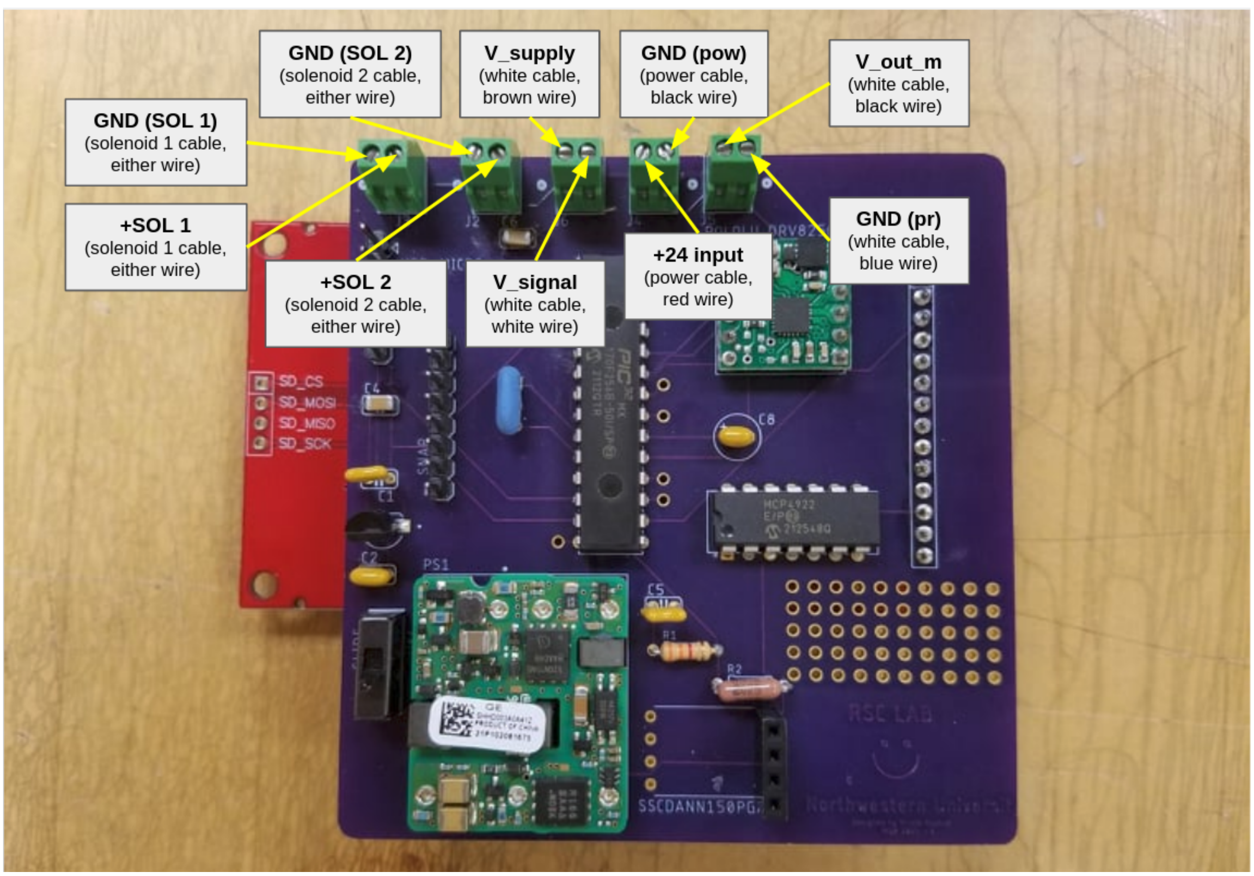


Figure 3.2.1: *Image of screw terminal connections.*

Note: V_supply and V_signal are the supply voltage and analog input signal (to control the pressure output) of the pressure regulator. V_out_m is the output from the monitor.

Please note that to remove any of the tubing, the barbed component of the connector must be pushed in and held in place as seen in the Figure 3.2.2 below.

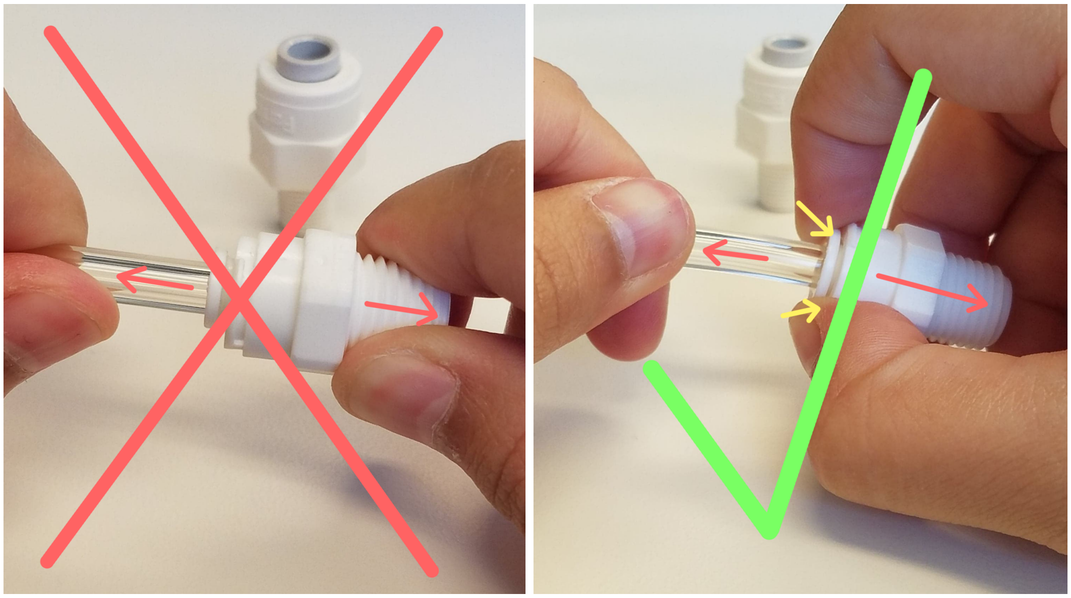


Figure 3.2.2: *Instructional image of how to remove tubing.*

Figure 3.1.3: *Image of all system connections.*

Software

4.1 IDE & Debugger

The following software is required to program the PIC32:

- Native C Compiler
  - MinGW (Windows)
  - gcc (Linux)
- Make

    - Obtained via MinGW (Windows)
    - make (Linux)
- Microchip XC32/32++ Compiler
- FTDI Virtual COM Port Driver (Windows)
    - All FTDI devices now supported in Ubuntu 11.10, kernel 3.0.0-19
- Terminal / Serial Interface
    - PuTTY (Windows)
    - PuTTY (Linux)

4.2 Control Library

Version Control

A git repository was created to develop the controller and to store component data sheets. Each controller feature has its own dedicated branch, and once a feature was developed it was merged with the master branch.

Source Files

adc.c - Contains the necessary functions to initialize and use the ADC and CTMU peripherals.

`void adcConfigureManual()`

Input:

N/A

Function:

Configures the ADC for manual sampling, sampling rate configured here.

`void adcConfigureManual()`

Input:

`adcPINS`: pins to use for scan mode

Function:

Configures the ADC for automatic sampling. **Double check AD1CON2SET**! The buffers need to be set manually, a consideration for future updates.

`int val = analogRead(char analogPIN)`

Input:

`analogPin`: PIC32 input pin

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually.

`int val = analogRead_auto()`

Input:

N/A, pins that are automatically sampled are pre-assigned

Output:

`val`: integer value from ADC buffer between 0-1023, sampled automatically.

`void ctmu_setup()`

Input:

N/A

Function:

Configures the Charge-Time-Measurement-Unit (CTMU) module on the PIC32. The module needs time to charge its capacitor.

`int val = ctmu_read(int pin, int delay)`

Input:

`pin`: PIC32 input pin

`delay`: time delay, on order of 10e5

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually.

`int val = av_cap(int pin, int delay, int win)`

Input:

`pin`: PIC32 input pin

`delay`: time delay, on order of 10e5

`win`: window of samples to average over

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually, averaged over the window provided. This is a workaround to calling the ctmu_read function recursively.

ili9341.c - Initializes the screen, as well as initializes serial peripheral interface (SPI) communication. There are also helper functions to display characters, draw, and clear the screen. (0,0) on the screen is the upper left corner unless otherwise defined/oriented.

`void SPI1_init()`

Input:

N/A

Function:

Initializes SPI module on PIC32. **This is where serial data out (SDO) and serial data in (SDI) are set.**

`void LCD_init()`

Input:

N/A

Function:

Sends commands via SPI to initialize the screen.

`void LCD_drawPixel(unshort x, unshort y, unshort color)`

Input:

x:	x screen coordinate
y:	y screen coordinate
`color`:	color, as defined in ili9341.h

Function:

Draws a pixel at (`x`, `y`) location in `color`, per ili9341.h.

`void LCD_clearScreen(short color)`

Input:

`color`:	color, as defined in ili9341.h

Function:

Defines color for entire screen.

`void print_char(unsigned short x, unsigned short y, char ch)`

Input:

x:	x screen coordinate
y:	y screen coordinate
ch:	ASCII character

Function:

Draws ASCII character `ch` at location (`x`, `y`).

```
void clear_space(unsigned short x, unsigned y, unsigned short
    end)
```
Input:

x:	x screen coordinate
y:	y screen coordinate
end:	length to clear

Function:

Clears ASCII chars.

```
void write_screen(int x, int y, char *msg)
```
Input:

x:	x screen coordinate
y:	y screen coordinate
`msg`:	message to write to screen

Function:

Writes message to screen by calling print_char.

init.c - Contains the PIC's configuration settings and contains functions to run through the initialization routines.

```
void init_pic()
```
Function:

Initializes core PIC32 configurations. **sysclk is configured here.**

uart.c - Initializes UART serial communication and contains the read/write functions.

```
void initUART(int desired_baud)
```
Input:

`desired_baud`: Baud rate for serial communication

Function:

Configures the UART module of the PIC32. **Tx/Rx pins are set here.**

```
void writeUART(char * string)
```
Input:

`string`: Message to be sent

Function:

Sends message via serial transmission.

`void readUART(char * message, int maxLength)`

Input:

`string`: Message to be sent

`maxLength`: Maximum length of message

Function:

Receives message via serial transmission.

main.c - Contains the main loop function, the transfer function for the pressure sensor, and some helper functions

`void ui()`

Function:

Displays a decorative header.

`void heartbeat(int freq)`

Input:

`freq`: Flash frequency

Function:

Flashes single pixel at (0,0). Should be rewritten as a timer/interrupt.

`double val = transfer_function(int voltage)`

Input:

`voltage`: 0-1023 value as returned by the ADC

Output:

`val`: output value is proportional to the applied pressure and atmospheric pressure.

The above proportional transfer function (*transfer_function*) was written specifically for this project by Alexander Hay in lieu of Honeywell's provided function, as it provided expected values under known conditions, while the Honeywell function did not. The pressure sensor was checked against a pressure gauge on the compressor air supply and the pressure gauge on the compressor tank.

Discussion & Future Work

During development, a significant amount of time was spent troubleshooting hardware. As such, I have compiled some notes of “things you might not know”, as much of the information required for certain devices were either vague or not available at all in the provided documentation. I’ve also included “a brief guide to desoldering when your component is stuck and driving you up a wall”, which is a collection of desoldering tips I’d recommend for new engineers, as this project requires a lot of soldering. Tips in designing PCBs and features that could prove useful are also included with the gerber files. (Please see appendix for more details.)

For the future, a thorough characterization of the forces applied by the cylinder onto the fingertip would be ideal to ensure a reliable, controlled force for every trial. Using a hand interface to mount the cylinder would help to standardize the distribution, and it would also keep the cylinder stationary.

The ability to control the airflow through the top opening of the cylinder could also help to maintain a more stable pressure value. While it may not have a significant effect on the study, adding an additional pressure sensor to measure the pressure at retraction could complement the pressure readings at extension (when the piston is exposed). Such measurements could also tell us what amount of force is necessary to force the cylinder into a retracted position while airflow is directed to the bottom chamber without requiring the test subject to overexert themself. Instead, the collected data could be used to determine an output pressure that would be challenging enough for the test subject to oppose without the concern of any harm.

Finally, additional sensing could be added to expand the measured data from each trial. Types of sensors that come to mind are a force sensor and a position sensor to further characterize the tactile stimulus provided. These sensors could also aid in the standardization of the setup, as previously mentioned with the hand interface. As mentioned in the previous iteration of this project, knowing if and when the piston engages with the finger is also important information, especially as pressure data is being logged in real time. To collect this data, a conductive and MRI-compatible material, such as aluminum foil, which is non-ferromagnetic, could serve as a capacitance sensor and safely be implemented to determine the exact moment of contact.

References

[1] Netta!

[2] Ahalya!

[3] Alex!

[4] Matt!

Appendix

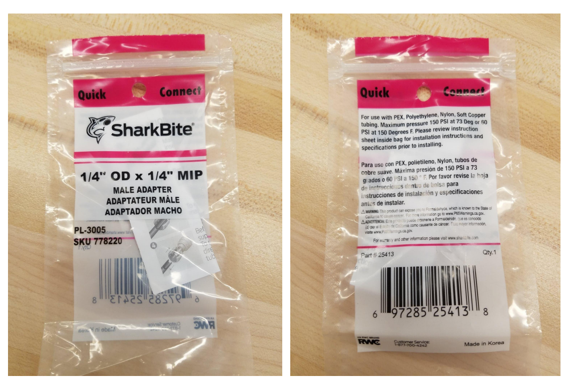
Bill of Materials

	Item Description	Qty	Item Number	Source
1	1-1/4" Diameter Chemical-Resistant Slippery PTFE Rod (1')	1	8546K17	https://www.mcmaster.com/8546K17/
2	1/8" ID, 1/4" OD Firm Polyurethane Tubing for Air and Water	1	5648K74	https://www.mcmaster.com/5648K74-5648K611/
3	2.62 mm Wide, 17.86 mm ID Oil-Resistant Buna-N O-Ring (pack of 25)	1	9262K974	https://www.mcmaster.com/9262K974/
4	1/4" Outside Diam, 1/8 NPTF, Polypropylene Push-to-Connect Tube Male Connector	2	78058690	https://www.mscdirect.com/product/details/78058690?fromRR=Y
5	¼” FNPT - ¼” Tru-Flate design T-style Plug	1	S-784	Purchased from Lemoi Ace Hardware[1]
6	SharkBite ¼” OD x ¼” MIP Male Adapter	1	25413	Purchased from Lemoi Ace Hardware[2]
7	California Air Tools Light & Quiet 1P1060S Portable Air Compressor	1	1P1060S	https://www.californiaairtools.com/ultra-quiet-series-of-air-compressor-contractor-grade/3-5-hp-air-compressor/cat-1p1060s/

TO BE CONTINUED USING SPREADSHEET SENT TO CTI

1:

2:



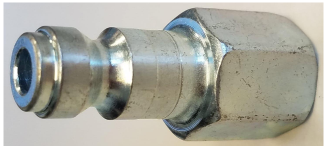


Figure 1. Tru-Flat Plug

Figure 2. SharkBite Adapter

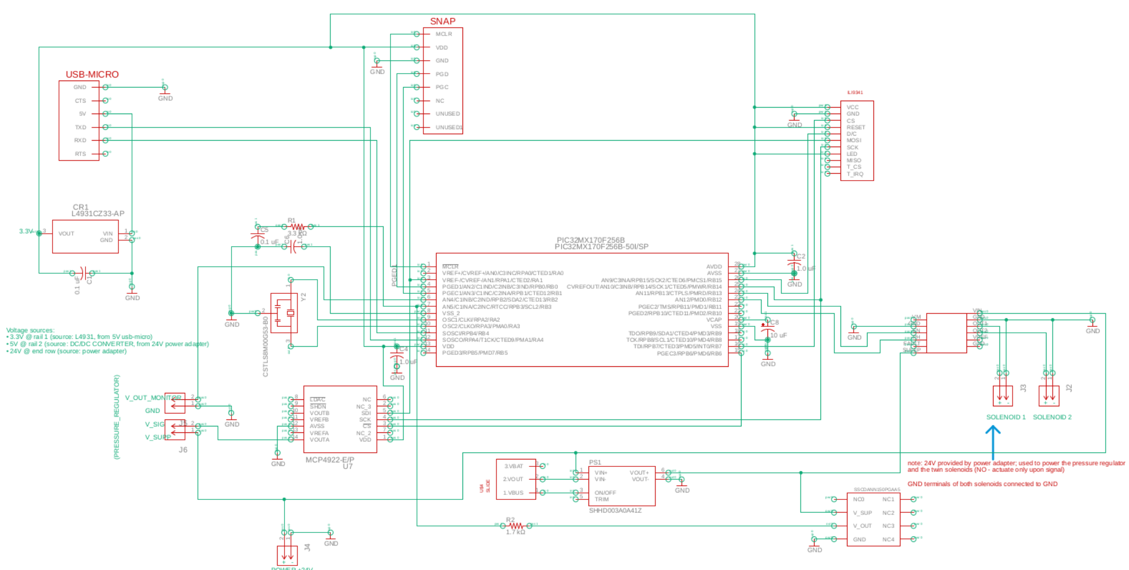
Code

Please see full source code on the github repository here.

Gerber Files

Please see the zipped gerber files for PCB ordering on the github repository here.

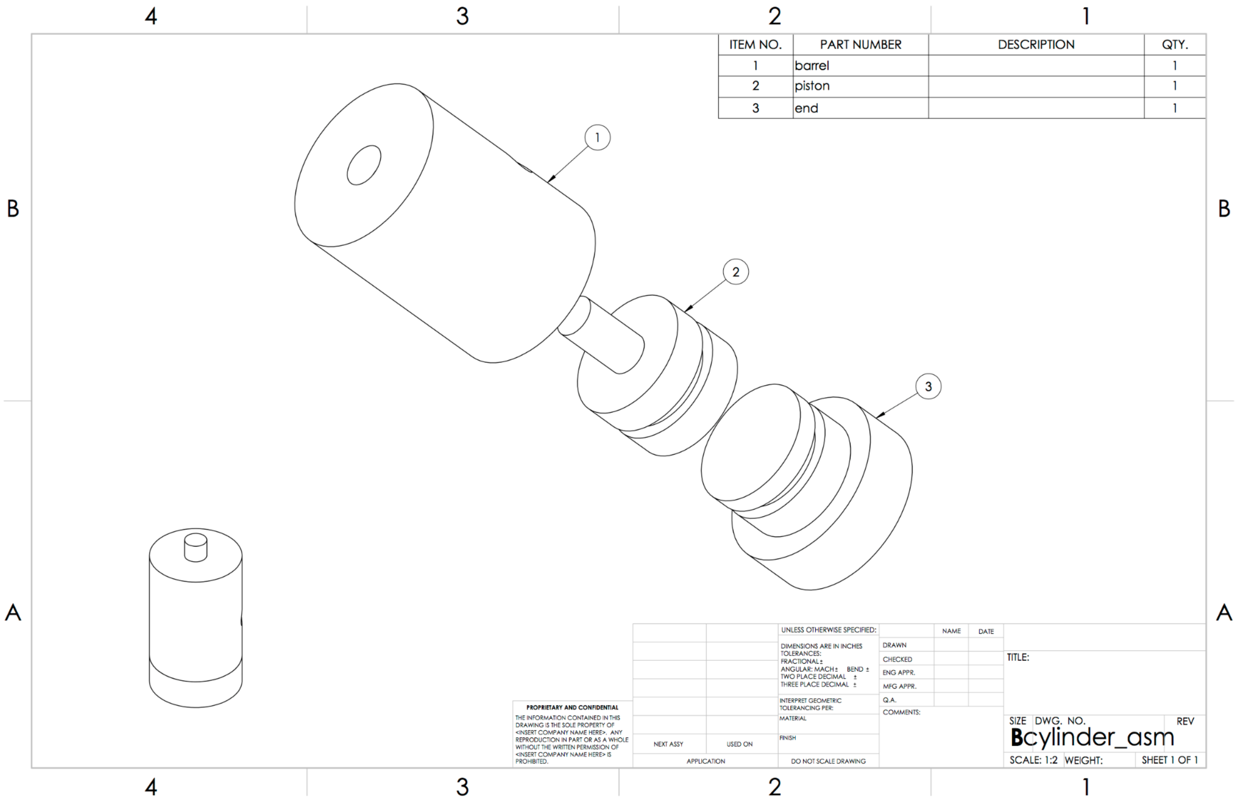
Circuit Overview



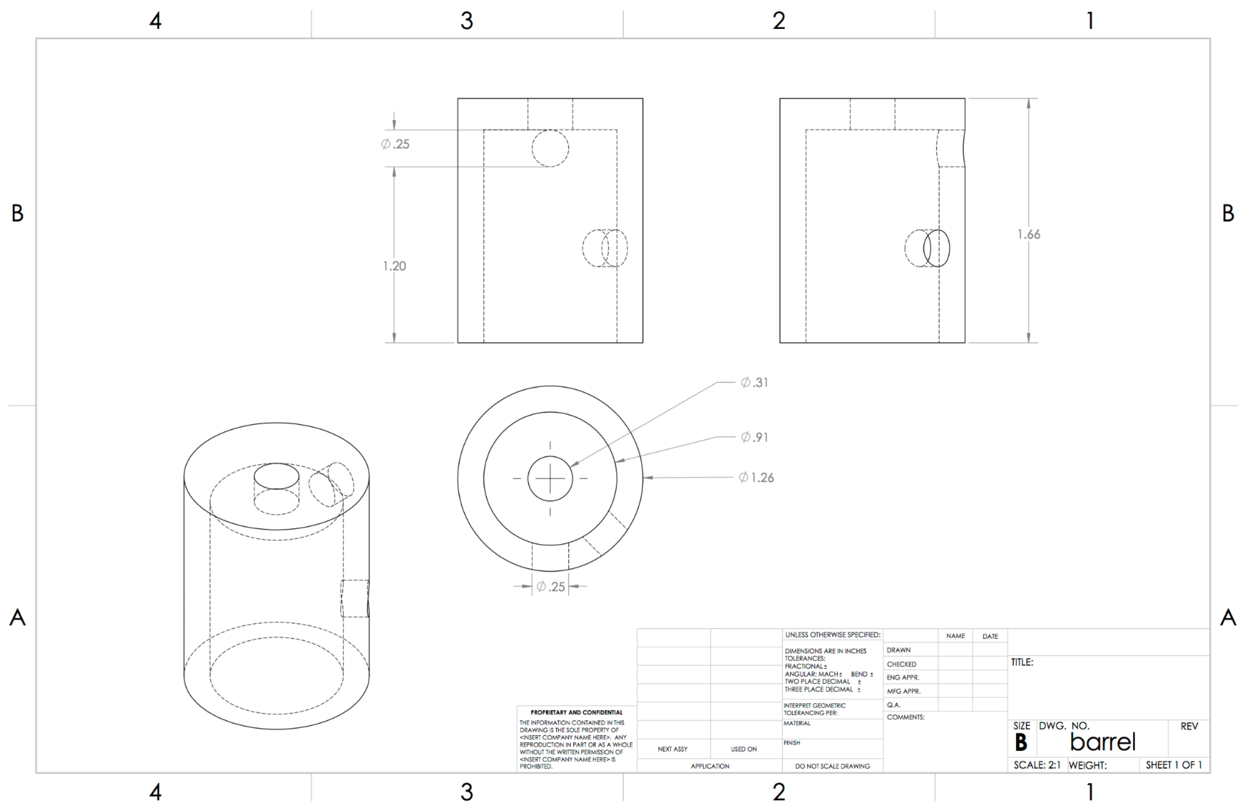
PIC32 - microcontroller; the brain	MCP4922 - digital to analog converter (for screen output)	DRV8256E - H-bridge to control the solenoid valve, pressure regulator	DC DC converter (SHHD…) - convert 24V source to 5V	ILI9341 (screen) - for easy debugging and aesthetics
CSTLS8M00G53-B0 - control timing	L4931 - voltage regulator, 5V to 3.3V	Slide switch - quick on/off switch	Capacitors - stores/releases charge	Resistors - electrical resistance
SNAP - PIC32 programming	USB - power, serial communication	Terminal blocks - wire connection	SSCDANN… - pressure sensor	***Please see schematic for pins***

CAD Drawings

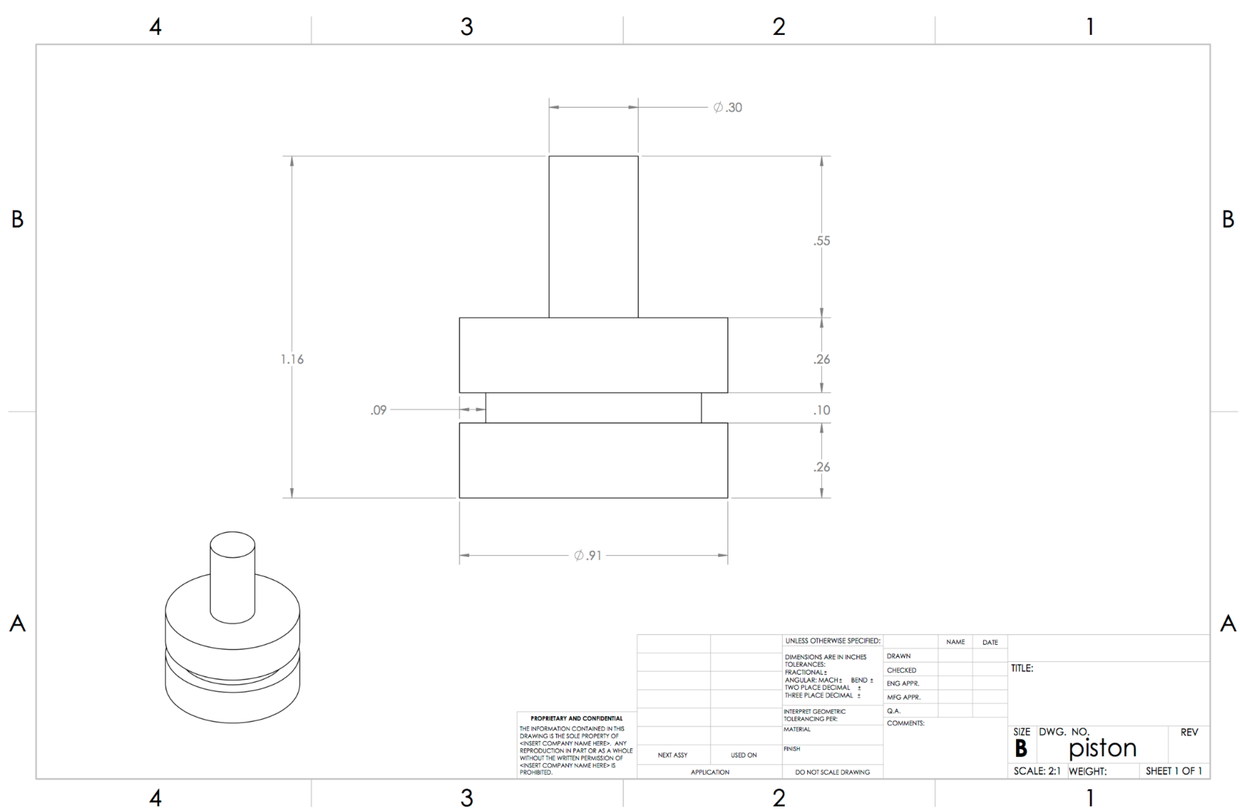
Assembly of the cylinder



Cylinder barrel



Cylinder piston



Cylinder end cap