

The Pennsylvania State University
The Graduate School
Department of Mechanical and Nuclear Engineering

**MECHATRONIC DEVICES
FOR FABRICATION, TESTING, AND IMPLEMENTATION
OF ROLLED ACTUATORS IN A COMPUTER-CONTROLLED BRAILLE DISPLAY**

A Thesis in
Mechanical Engineering
by
Michael Robinson

© 2012 Michael Robinson

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

May 2012

The thesis of Michael Robinson was reviewed and approved* by the following:

Christopher D. Rahn
Professor of Mechanical Engineering
Thesis Co-Advisor

Qiming Zhang
Distinguished Professor of Electrical Engineering
Thesis Co-Advisor

Sean Brennan
Associate Professor of Mechanical Engineering

Karen A. Thole
Professor of Mechanical Engineering
Head of the Department of Department or Graduate Program

*Signatures are on file in the Graduate School

ABSTRACT

Specialized devices are needed to meet the unique challenges of manufacturing, testing, and implementing rolled, core-free P(VDF-TrFE-CFE) actuators for Braille cell applications. Actuators are rolled to provide structural support, and designed without cores to simplify the final device. This thesis outlines the design, implementation, and experimental testing results of three mechatronic devices. First, a compact and easy-to-use machine was designed and built to assist with rolling thin films into cylindrical actuators. Second, in order to confirm that the Braille actuators met the .5 N force requirement of most refreshable Braille displays, a computer-controlled force and strain testing device was designed that facilitated an investigation into the buckling characteristics of these actuators. An average buckling load of 1.84 N was found for a sample of 10 actuators, reasonably close to theoretical predictions of an Euler-Bernoulli buckling model. Last, a handheld, USB-powered Braille display was produced and experimentally tested that allowed a user to receive Braille characters from a computer.

TABLE OF CONTENTS

| | |
|--|------|
| LIST OF FIGURES | v |
| LIST OF TABLES | vii |
| ACKNOWLEDGEMENTS | viii |
| Chapter 1 Introduction | 1 |
| Chapter 2 Actuator Rolling | 3 |
| Design | 4 |
| Results | 5 |
| Chapter 3 Mechanical Testing | 7 |
| Design | 8 |
| Validation | 10 |
| Experimental Methods | 13 |
| Euler Buckling Model | 15 |
| Experimental Results | 17 |
| Chapter 4 Handheld Braille Display | 20 |
| Mechanical Design | 20 |
| Electrical Design | 22 |
| Embedded Software | 25 |
| User interface and Digital Communication | 25 |
| Testing | 26 |
| Chapter 5 Conclusions and Future Work | 31 |
| References | 32 |
| Appendix A: Force and Strain Testing Machine LabVIEW Interface | 34 |
| Appendix B: Force and Strain Testing Machine Embedded Software | 36 |
| Appendix C: Force and Strain Testing Machine Circuit | 36 |
| Appendix D: Handheld Braille Cell Embedded Software | 45 |
| Appendix E: Handheld Braille Cell Circuit | 51 |

LIST OF FIGURES

| | |
|--|----|
| Figure 2-1: Photograph of the actuator rolling machine | 5 |
| Figure 2-2: Actuator rolling machine controls (a) and demonstration of manual table positioning (b)..... | 5 |
| Figure 2-3: Photograph of a finished actuator made with the actuator rolling machine | 6 |
| Figure 3-1: Photograph of the force and strain testing machine with part descriptions. Note that the top vice is held fixed during testing and the lower vice rises to compress the sample. | 8 |
| Figure 3-2: Force and strain testing machine positioning validation. The solid blue line is the commanded displacement, and the green line with crosses is the displacement measured with the laser vibrometer. (a) shows the displacement over a series of 10 steps and (b) shows an enlarged version of the region inside the dashed box. | 11 |
| Figure 3-3: Force and strain testing machine stiffness measurement. The blue solid line shows the measured data and the green dashed line indicates the best linear fit. The slope of the linear fit is used as the device stiffness..... | 12 |
| Figure 3-4: Stiffness measurement validation using multiple compression and relaxation cycles of a spring with a known spring constant. The solid blue line shows the measured data and the green dashed line indicates the best linear fit. | 13 |
| Figure 3-5: Image (a) shows the experimental setup for buckling testing with important features labeled. Image (b) shows a CAD model of the experimental setup with the support structure rendered transparent and the test actuator shown in blue and green. The direction of movement of the lower vice is also indicated..... | 14 |
| Figure 3-6: (a) Buckling results for three sample actuators. (b) Three experimental runs with the same actuator indicating the first run (blue solid line), the second run (green line with crosses) and the third run (red dot-dash line)..... | 18 |
| Figure 3-7: Actual and predicted buckling loads plotted against actuator stiffness. Experimental results (blue x), calculated load for first buckling mode (solid green line), calculated load for second buckling mode (red dashed line), and calculated load for best fit model (light blue dotted line) | 19 |
| Figure 4-1: Images illustrating the mechanical features of the Braille cell: (a) completed Braille cell,..... | 21 |

| | |
|--|----|
| Figure 4-2: (a) Conductive fabric being tested in an experimental Braille cell. (b) Raised features are used to keep the conductive fabric under one pin from interacting with its neighbors. | 23 |
| Figure 4-3: Main PCB of the handheld Braille display with part descriptions | 24 |
| Figure 4-4: Actuator charging and discharging circuit schematic | 25 |
| Figure 4-5: Computer interface for handheld Braille display. | 26 |
| Figure 4-6: Plot (a) shows that the output voltage decreases in response to activating additional charging circuits. Plot (b) shows the relationship between input current and the number of active charging circuits. | 27 |
| Figure 4-7: Braille pin displacement validation using a laser vibrometer. | 28 |
| Figure 4-8: Plot (a) shows actuator displacement for three different duty cycles at 50 Hz PWM frequency. The blue solid line is 25 percent duty, the green dashed line is 50 percent and the red dot-dashed line is 100 percent. Plot (b) shows the displacement profile of an actuator with a 50 percent duty cycle and 50 Hz switching frequency. Plot (c) shows the displacement profile of an actuator with a 50 percent duty cycle and a 10 Hz switching frequency. It can be seen that increasing the switching frequency significantly reduces displacement fluctuations..... | 29 |
| Figure 4-9: Image (a) shows the Braille cell without any actuation voltage and (b) shows the cell displaying the letter “k”, which corresponds to pins 1 and 3 raised..... | 30 |
| Figure A-1: Force and strain testing machine user interface. | 34 |
| Figure C-1: Circuit schematics for force and strain testing machine. | 44 |
| Figure D-1: Standard Braille pin numbering. | 45 |
| Figure E-1: Handheld Braille display circuit schematic. | 51 |

LIST OF TABLES

Table 3-1: Actuator buckling results.....17

ACKNOWLEDGEMENTS

I would first like to thank my wife, Gabrielle Robinson, for all of the love and support she has given me over these two years. Her incredible ability to help me both relax and stay focused was a critical part of all that I have accomplished. She is my continual motivation to reach for more, and I will always be grateful for that.

I would also like to sincerely thank my advisor, Dr. Chris Rahn, for allowing me to be a part of this project. His advice was crucial to the successful completion of this project and I greatly appreciate the research atmosphere he has fostered in the lab. I also want to thank him for his support of my spending some of my time pursuing educational research.

Third, I want to thank Dr. Qiming Zhang for his support and advice on this project. I also appreciate the assistance that the members of his lab, specifically Yu, David and Will, gave me by providing me actuators to test and information about materials.

Lastly I want to thank all of the members of the Mechatronics Research Lab. I've enjoyed the time we have spent together over the last year and a half, and I wish you all the best.

Chapter 1

Introduction

There has been a growing interest recently in using electroactive polymer (EAP) actuators in refreshable Braille displays [1],[2]. Computer-controlled Braille displays allow the blind to interact with modern computer technology; however, conventional displays use bulky and expensive actuators which limit the number of characters that can be displayed at one time and the minimum device size. EAP actuators on the other hand are attractive because they can meet the requirements of Braille displays, specifically that the actuator should extend at least .5 millimeters and produce at least .5 Newtons of force, while reducing overall device size and complexity. While the possibilities of these actuators are exciting, there are many challenges associated with their fabrication, reliability and implementation.

Several groups have investigated the possibility of using alternative actuation techniques in tactile Braille displays. Choi et al. investigated using diaphragm-based actuators and created a compact, multi-character display [3]. Kato et al. used polymeric actuators to create a lightweight and flexible sheet display which allowed many characters to be placed in a small space [4]. Spinks et al. investigated using polypyrrole actuators to create small, multi-character displays [5].

P(VDF-TrFE-CFE) blend materials are attractive options for Braille cell actuation because the films have shown relatively high strain (>5%) at a modest field (150 V/ μm) and have a high Young's modulus (>.3GPa) [6]. Previous investigations on actuator fabrication and testing concluded that rolled, core-free P(VDF-TrFE-CFE) actuators could meet Braille displacement requirements [7], [8]. An experimental single character Braille cell was also fabricated using these actuators.

Although a complete description of the manufacturing process for these actuators can be found elsewhere [8], a summary will be provided here for completeness. Material films are

produced by a process of solution casting followed by stretching and annealing. Next, the films are attached to metal frames and electrodes are deposited. The single layer films are removed from the frames and laminated together. This bilayer film is then rolled up into a cylinder and trimmed to an appropriate length. Finally, steel balls are glued onto the ends of the cylinder with a conductive epoxy to create electrical connections.

This thesis presents the designs and results of three mechatronic devices which were created to address the challenges of fabricating, testing and implementing P(VDF-TrFE-CFE) actuators. First, a rolling machine was developed to replace a semi-automated process used by previous investigators [7]. It was found that this relatively simple and inexpensive device can reliably and conveniently roll actuators. Second, prior investigations reported that some actuator samples failed during blocked force tests, but the mechanism responsible for these failures could not be reliably identified. To address this issue, an experimental setup was created to externally strain actuators and measure the resultant forces. This device was used to determine the assembled stiffness of individual actuators and to determine the critical buckling load of actuator samples. An Euler buckling model was developed with these data to predict actuator buckling loads. Finally, a compact, computer-controlled Braille display was created that received power and communication over a standard USB port.

Chapter 2 Actuator Rolling

Rolling EAP actuators from a flat film into a cylinder presents unique challenges. Thin films (typically around 11 micrometers of bilayer thickness) are used to reduce the voltage required to meet the Braille display standard of .5 millimeters of displacement; however, these films lack the ability to support significant loads, such as the .5 Newtons of force required in refreshable Braille displays, and are rolled into cylinders to improve structural characteristics. There are several conditions that need to be maintained to produce an acceptable final actuator. First, the film needs to be kept in constant tension while it is rolled in order to ensure that the final actuator has tightly wound layers. Second, proper alignment must be maintained during the rolling process. Lastly, the film should be kept smooth, because wrinkles created during rolling will likely cause the finished actuator to fail. All of these factors make hand rolling very difficult, and motivate the development of a specialized device to assist with this process.

Previously, a semi-automated machine was developed to roll these actuators [7]. In this setup, the user needed to align the film on a paddle, and the machine would handle the remainder of the rolling steps. Although this system was effective, there were a number of factors that motivated the development of a new solution. It was desirable to create a more flexible design so that small changes in actuator dimensions didn't require a major equipment redesign. Also, the text based interface used by the semi-automated machine led to a significant learning curve and operators found it difficult to use. Finally, the fully automated device was relatively large, and it was desirable to reduce the footprint size.

A new rolling machine was designed with the intent of augmenting the skill of the operator, instead of attempting to replace the operator entirely. Actuator manufacturing needs to be done in a clean-room environment where space is typically scarce which dictated that the new machine be small, easy to move, and easy to store. User interactions with this machine were also

designed to be as intuitive as possible so that the operator can focus his or her attention on the rolling task.

Design

A picture of the final device can be seen in figure 2-1. A sliding clamp holds the film in place while a suspended weight provides even tension on the film. A mass of roughly 4 grams was found to be sufficient; however, this mass can be easily changed if needed. The film is glued with a removable adhesive to a mandrel and a stepper motor rotates the mandrel. An Arduino microcontroller board provides control signals to the stepper motor driver. The mechanical parts of this machine were produced by water jet cutting or by fused deposition modeling rapid prototyping.

User interactions were eased by reducing the overall number of controls and by designing with the user-centered principles of affordances and natural mappings [9]. Affordances are aspects of a design that help tell a user what actions are possible, and natural mappings are created when the relationship between device inputs and outputs matches the intuition of a user. For example, the controls on this unit make their function obvious, and provide an effective affordance. Figure 2-2 shows the user controls implemented on this device. A natural mapping is used to make adjusting machine speed as intuitive as possible. Turning the knob to the right increases the mandrel's rotating speed while turning the knob to the left decreases its speed. Even without instructions, a user could quickly understand how to operate this device. The film table on this device also provides the user with an affordance. The table can be tilted by hand to bring the actuator film in contact with the rolling mandrel. The table is moved by hand into contact with the actuator at the end of the rolling process to keep the film from unraveling. This allows the operator to manually adjust the force.

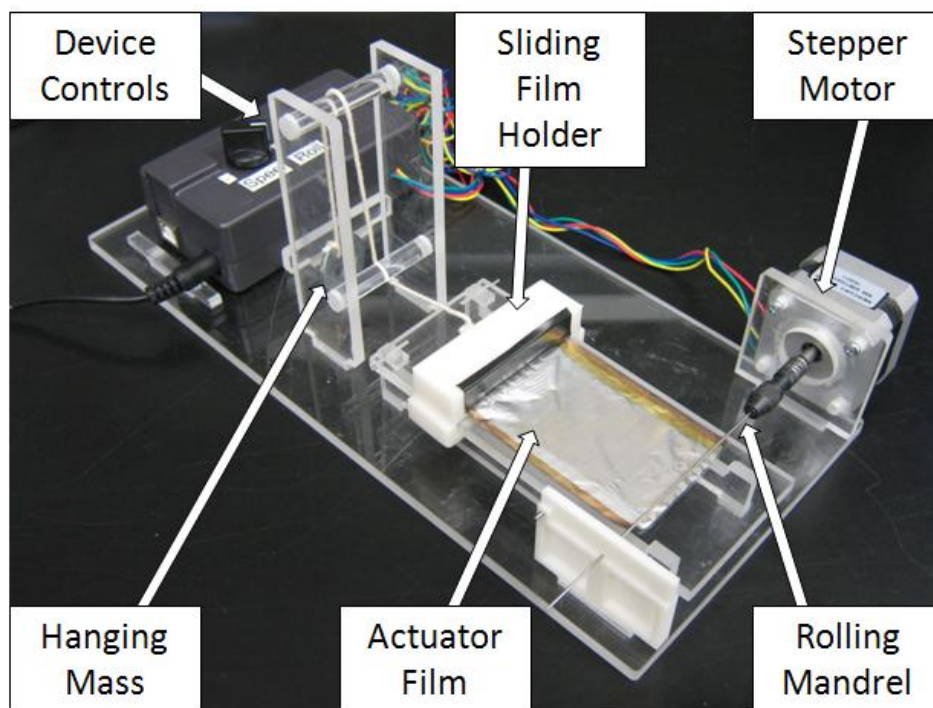


Figure 2-1: Photograph of the actuator rolling machine

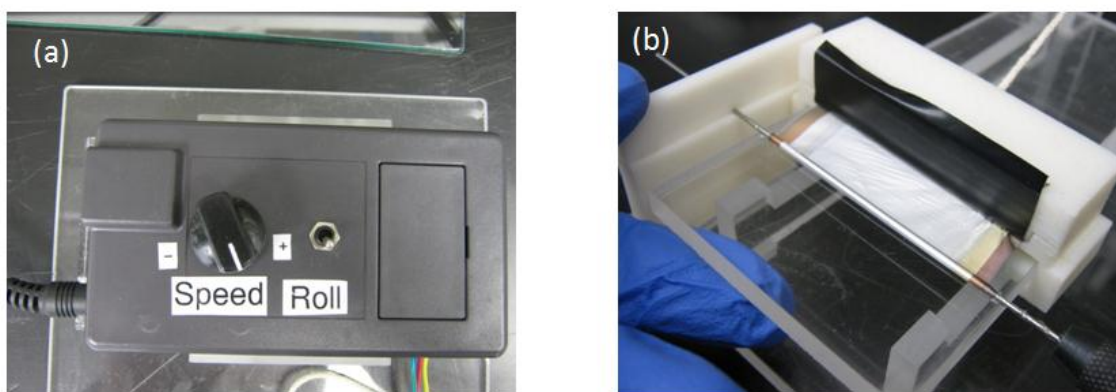


Figure 2-2: Actuator rolling machine controls (a) and demonstration of manual table positioning (b)

Results

Two different actuator lengths were used with this machine. One set of films were 75 millimeters in the direction of rolling and 45 millimeters wide. A second set of films were also 75 millimeters in the direction of rolling but were roughly 50 millimeters wide. A 1 millimeter diameter mandrel

was used for both actuator sizes. Figure 2-3 shows an example finished actuator made with the 45 millimeter wide film. Once the ends of the cylindrical film were trimmed and steel electrodes were attached, the finished actuator was roughly 40 millimeters in length.



Figure 2-3: Photograph of a finished actuator made with the actuator rolling machine

Actuators made with this device were found to have a smaller diameter than those made with the semi-automated machine. A previous study reported an average actuator diameter of 2.3 millimeters [7]. A set of 6 actuators made with this new device was found to have an average diameter of roughly 1.8 millimeters, an advantage in the Braille cell application. Members of the actuator fabrication team also preferred to use the new rolling machine, because it was more convenient and it allowed them to roll actuators in less time than was previously required. Typically, an actuator can be rolled in less than five minutes on this machine. Six actuators are required to form each Braille character in the final display.

Chapter 3 Mechanical Testing

The long and slender shape of the rolled Braille cell actuators motivated an investigation into their buckling characteristics. The dimensions of these actuators were largely determined by the standard Braille requirements, specifically the requirement that the Braille pins must move at least 0.5 millimeters and adjacent pins within a character must be spaced 2.5 millimeters apart [10]. The overall length of these actuators was dictated by the displacement requirement because the displacement produced by these actuators is proportional to active length. The outer diameter is limited by the Braille pin spacing requirements and the practical manufacturing constraints on how closely holes in a final Braille cell could be placed. Typical actuators are 40 millimeters in length and 1.8 millimeters in diameter.

Previous investigations revealed that some samples underwent a sudden and significant reduction in force during blocked force testing; however, a definitive conclusion about the mechanism responsible for this failure could not be reached [7], [8]. While some actuators exhibited behavior that appeared to be buckling, other actuators were able to reach forces that were a factor of two or greater without buckling. Previous tests were also electrically actuated, which made it difficult to rule out an electrical explanation for these failures. To investigate the buckling properties of these actuators, a device was created that could apply an external displacement to a sample actuator and measure the resulting forces. The data produced by this machine was used to determine the maximum buckling load and the assembled actuator stiffness. These stiffness values were used with an Euler buckling model to predict actuator failure.

Design

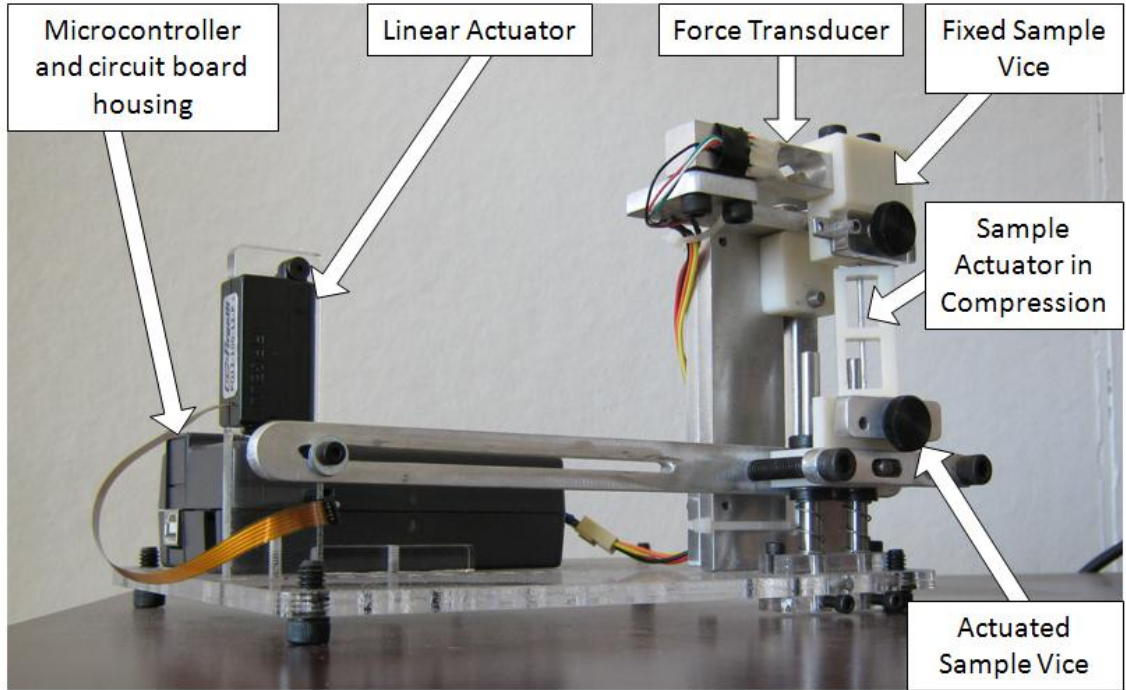


Figure 3-1: Photograph of the force and strain testing machine with part descriptions. Note that the top vice is held fixed during testing and the lower vice rises to compress the sample.

Tradeoffs between accuracy and cost determined the final design of this machine.

Researchers from material science, electrical engineering and mechanical engineering met before any design work was started, and determined that this device should be able to accurately produce and measure displacements of at least 1 millimeter with a precision of 20 micrometers or less.

The device also needed to measure force to within .05 Newtons. Due to the small displacements required, a Firgelli PQ12-100-12-P linear actuator was selected, which had a precision of 100 micrometers, and a mechanism was created to improve this precision to meet the testing requirements. These features can be seen in figure 3-1. By incorporating a sliding joint on the linear actuator mount and on the sample vice, a linear relationship was created between actuator displacement and clamp displacement. In this experiment, the sample clamp moved roughly 1/9th

the displacement of the linear actuator. Two springs were placed below the sample vice to eliminate mechanical backlash produced by the sliding joints.

Mechanical parts for this device were created by using water jet machining, CNC laser cutting or fused deposition modeling. The height of the upper clamp could be adjusted using thumb screws on the back of the unit to accommodate different sample lengths. Also, the linear actuator could be repositioned horizontally to change the length of the lever arm between the linear actuator and the sample vice. This would allow for samples with greater displacement requirements to be tested in the future.

Displacement was measured with the internal potentiometer on the Firgelli actuator. This potentiometer had a stroke of 20 millimeters and the analog to digital converter used to read this sensor had a resolution of 10 bits. This provided a resolution of roughly 20 micrometers at the linear actuator. The mechanical reduction of the lever mechanism improved this resolution to 2.25 micrometers.

Force was measured with a Transducer Techniques EBB-1 load cell. This sensor had a rated capacity of 9.81 Newtons, which was sufficient to measure typical forces produced by these EAP actuators. An instrumentation amplifier amplified the signal from the load cell into a useful range for data collection. A final resolution of .02 Newtons was obtained with a 10 bit analog to digital converter.

In order to simplify device operation and data collection, a graphical user interface was created with National Instruments' LabVIEW. The displacement of the actuator vice could be controlled manually, or the device could be set to produce a ramp input at a user specified rate. Calibration settings could also be adjusted in the virtual instrument. Force and displacement results were presented to the operator in real time and could easily be exported for later analysis. Further details on the LabVIEW interface can be found in appendix A.

An Arduino Uno board was selected for linear actuator control and data acquisition. This board was relatively simple to interface with National Instruments' LabVIEW software, and also helped to keep overall device cost to a minimum. Communication was handled over a standard USB port. LabVIEW sent strings of delimited data to the Arduino controller, which parsed the strings to obtain data. A sampling rate of approximately 10 hertz was used. For a more thorough explanation of the embedded software used in this device, see appendix B. All measurement and control functions were implemented on the Arduino board. A custom circuit board was fabricated to hold the motor driver and signal conditioning circuitry for the load cell. Additional information and circuit diagrams can be found in appendix C.

Validation

Before actuator testing could begin, it was necessary to ensure that the measurements provided by this device were accurate. Proof masses were used to validate the force transducer. A calibration was performed every time the device was used to ensure that accurate results were obtained. It was necessary to allow the force measurement system to warm up for roughly thirty minutes before accurate measurements could be made. To provide an estimate of the uncertainty associated with the force measurement, the device was left unloaded for 30 seconds and data were recorded. The results were found to be roughly normally distributed with a standard deviation of .019 Newtons, indicating that there was acceptably low noise.

The platform positioning system also needed to be validated. This step was necessary to ensure that the positioning system was consistent, and that mechanical tolerances did not produce excessive measurement errors. Ten position step changes were commanded of the system, and the true displacement of the actuator vice was measured by a PolyTec OFV 534 laser vibrometer. The results of this test can be seen in figure 3-2. The average error was found to be -1.8 micrometers

with a standard deviation of 6.7 micrometers. The largest steady-state error recorded was 15 micrometers. These measurements confirmed that the positioning system was sufficiently accurate.

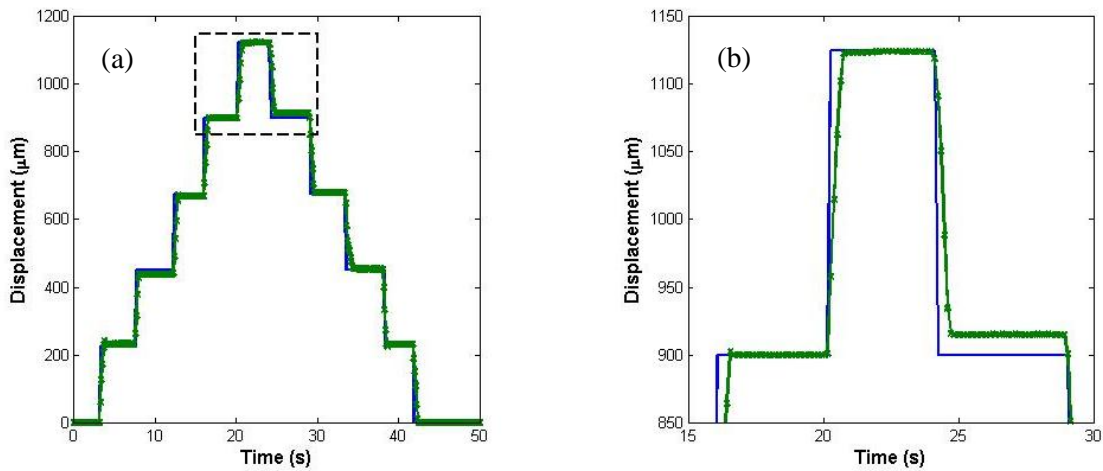


Figure 3-2: Force and strain testing machine positioning validation. The solid blue line is the commanded displacement, and the green line with crosses is the displacement measured with the laser vibrometer. (a) shows the displacement over a series of 10 steps and (b) shows an enlarged version of the region inside the dashed box.

In order to ensure precise stiffness measurements were obtained, the compliance of the measurement system needed to be taken into account. Any compliance in the testing equipment would act as a spring in series with the actuator, causing the measured stiffness to be less than the actual stiffness. To determine the measurement device stiffness, a piece of aluminum was placed in the sample vice which was large enough to have negligible strain at the testing forces. A linear regression was performed on several experimental runs, and an average stiffness value of .033 Newtons per micrometer was found. Since the sample was assumed to have no appreciable strain, this value was used as the device stiffness. A sample test result can be seen in figure 3-3. This stiffness is significant because it is roughly three times greater than the highest actuator stiffness, and would lead to significant errors if not accounted for. The actual sample stiffness is calculated from

$$\frac{1}{k_{sample}} = \frac{1}{k_{measured}} - \frac{1}{k_{device}}, \quad (1)$$

where k_{sample} is the sample stiffness, $k_{measured}$ is the measured stiffness found from the linear region of a force and strain plot, and k_{device} is the stiffness of the experimental measurement device.

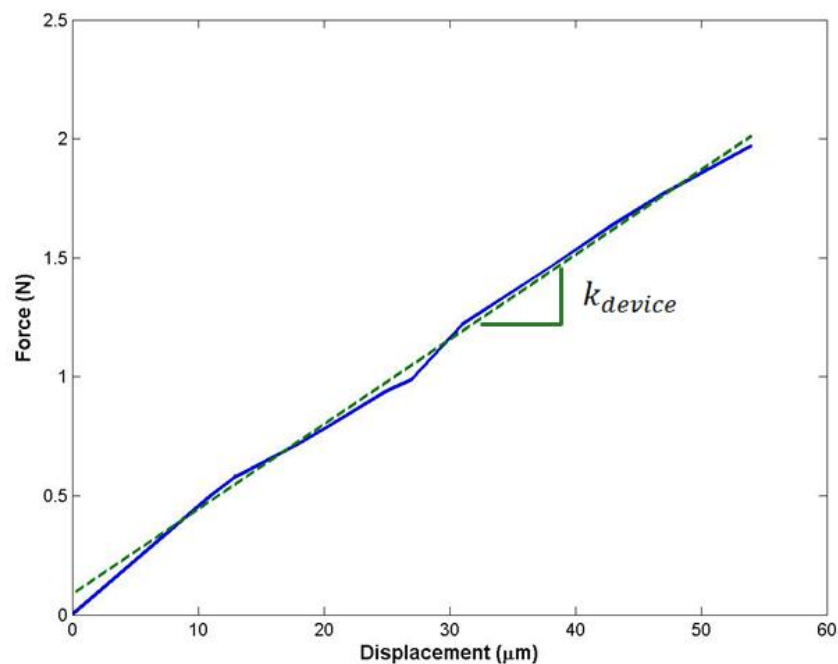


Figure 3-3: Force and strain testing machine stiffness measurement. The blue solid line shows the measured data and the green dashed line indicates the best linear fit. The slope of the linear fit is used as the device stiffness.

Sample stiffness measurements were validated using a spring with a known spring constant. This spring was placed between vices and was compressed and released for several cycles. The data from this test can be seen in figure 3-4. The spring constant reported by the manufacturer was .0032 Newtons per micrometer, and the stiffness found from these data after applying equation 1 was .0031 Newtons per micrometer.

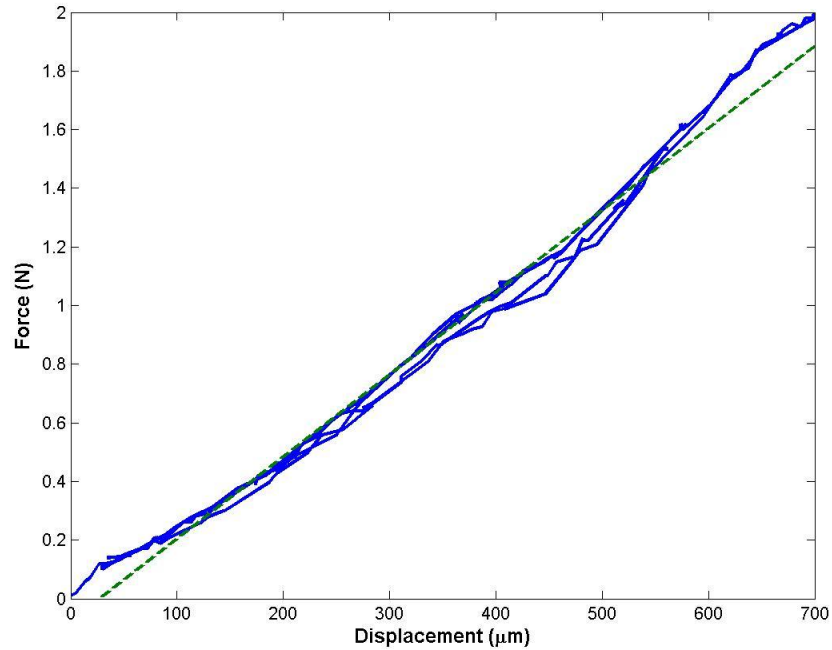


Figure 3-4: Stiffness measurement validation using multiple compression and relaxation cycles of a spring with a known spring constant. The solid blue line shows the measured data and the green dashed line indicates the best linear fit.

Lastly, it was desirable to evaluate the consistency of stiffness measurements over repeated tests. A sample actuator was cycled between 0 and 100 micrometers of displacement 10 times and linear fits were determined for each cycle. A small displacement was used to ensure that buckling did not occur. Linear regression was performed on the results of this test, and a slope was calculated for each compression cycle. The standard deviation of these data was found to be .00025 Newtons per micrometer which is roughly 2 percent of the typical actuator stiffness.

Experimental Methods

Sample actuators were placed in a support fixture and allowed to rest on the top surface of the lower vice. This fixture supported the ends of the actuator and the middle, but left the remainder of the actuator open. The justification for this design is explained below in the section

on the Euler buckling model. The upper vice was manually lowered until contact was made with the actuator. The experimental setup can be seen in figure 3-5. LabVIEW was then used to raise the lower vice at a rate of roughly 55 micrometers per second until buckling was observed. Force and displacement data were logged for later analysis.

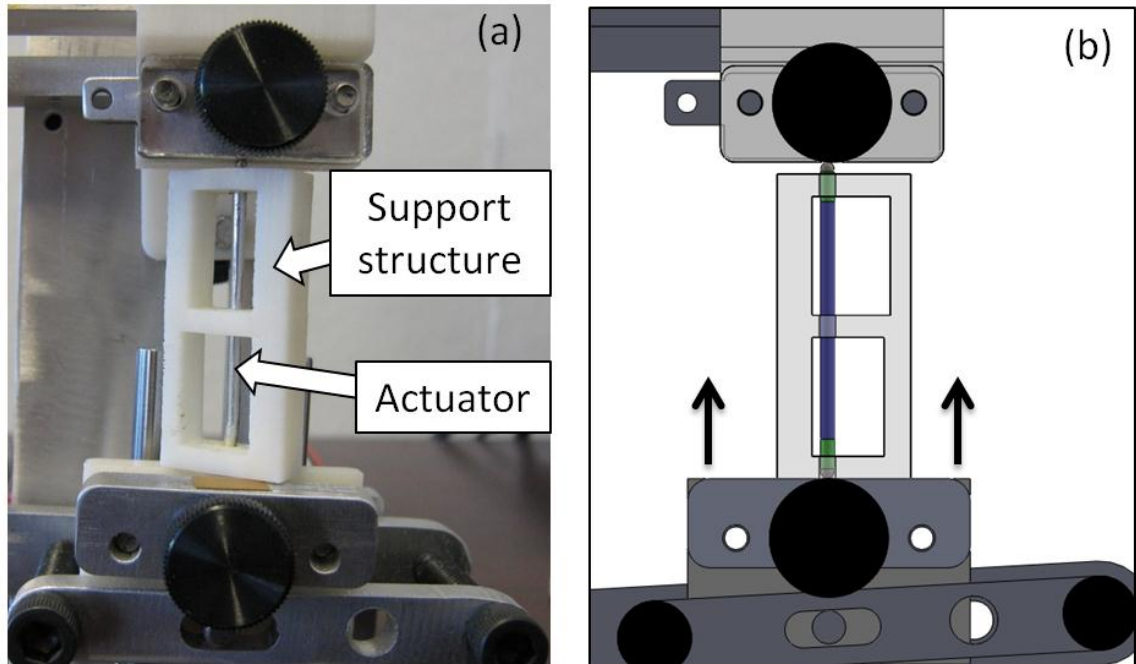


Figure 3-5: Image (a) shows the experimental setup for buckling testing with important features labeled. Image (b) shows a CAD model of the experimental setup with the support structure rendered transparent and the test actuator shown in blue and green. The direction of movement of the lower vice is also indicated.

Force and displacement data generated in this experiment provided two important pieces of information: the actuator's stiffness in compression and the buckling load. Actuator stiffness was determined by performing a linear regression on the initial linear region of the data. Equation 1 was then applied to account for the effect of measuring device stiffness. The highest force value recorded during the experiment was used as the buckling load.

Euler Buckling Model

Euler buckling has been proposed as a model for the failure of these actuators in compression [7]; however, there are several assumptions that need to be verified before this assumption can be used. The most important question is whether the cross section can be treated as solid. As discussed in chapter 2, these actuators are rolled into a cylinder, and the ends of the cylinder are held together with an adhesive, but it was not known whether the layers would act as a solid cross section especially under a compressive load. Second, the stiffness of the assembled actuator needs to be known to apply Euler buckling. The film itself is known to have a Young's modulus of roughly .3 GPa [6], but the final device has electrodes, glue, and steel balls, which made the assembled actuator stiffness unknown. Lastly, the manufacturing process for these actuators has a great deal of variability, and it was unclear how much affect this variability would have on buckling.

One goal of this experiment was to determine what support conditions were needed to prevent actuators from buckling at loads below .5 Newtons, which is the desired level for refreshable Braille displays [5]. A previous study found that actuators supported in a continuous tube failed at an average load of 1.54 Newtons [8]. This load is significantly higher than the Braille requirement and it was thought that there could be benefits to using different support structures even if the critical buckling load was slightly reduced. For example, if the actuators were only supported at the ends and in the middle, substantial material could be removed from the Braille display, saving both weight and cost. In order to investigate whether this method would provide adequate support, all actuators were tested in the fixture shown in figure 3-5 above.

The effective length of the actuators needed to be determined before an Euler buckling analysis could be performed. Effective length is proportional to the original length of the sample;

however, it is scaled by different factors depending on boundary conditions and the mode of buckling failure. It was not possible to find an exact analytical match for the true boundary conditions represented in this experiment, because the actuators were not perfectly constrained by the support fixture. The holes around the actuators needed to be slightly larger so that the actuator could move freely. Also, the glued ends of the actuators were typically larger than the actuator itself, forcing the holes to be slightly oversized. These tolerances allow the actuators to move a small distance laterally during testing. By observing the buckling behavior of these actuators, it was determined that the ends could rotate in the fixture, and pinned end conditions were appropriate.

The effective length of the actuator also depends on the buckling mode. The actuator supports were designed to force the actuators to fail in the second mode; however, the true buckling load was found to be substantially less. For this reason, predicted buckling loads are calculated for both the first mode and second mode using the Euler buckling model

$$P_{cr} = \frac{\pi^2 EI}{l_{eff}^2} = \frac{\pi^3 * k_{actuator} * l_0 * (d_o^4 - d_i^4)}{64 * A * l_{eff}^2}, \quad (2)$$

where E is the Young's modulus, $k_{actuator}$ is the stiffness of the actuator, l_0 is the initial length of the actuator minus the length of the steel electrodes, d_o is the outer diameter, d_i is the inner diameter, A is the cross sectional area, and l_{eff} is the effective length [11]. The cross sectional area of the actuator is calculated by multiplying the bilayer thickness by the length of the unrolled film in the rolling direction. If the bilayer thickness was not known, an average value of 11 micrometers was used in its place. The mandrel diameter of 1 millimeter was used as an approximate inner diameter. The overall length of the actuator was used as the effective length

(l_{eff}) while 4 millimeters were subtracted off the length to find l_0 . This was done to reduce the effect of the solid end electrodes on the theoretical buckling load.

Experimental Results

There was found to be a large amount of variability in the buckling load of the Braille cell actuators. Table 3-1 shows testing results for all 10 sample actuators and it can be seen that the highest buckling load was roughly 3.9 times greater than the smallest. In order to further demonstrate the variability in buckling behavior, Figure 3-6a shows the test data for three sample actuators. It should be noted that not only did actuators fail at different loads, but each sample had different behavior around the critical load. For example, some actuators showed a drastic reduction in load immediately following buckling, while other samples had a more gradual leveling off of force. Figure 3-6b shows three buckling cycles for the same actuator, and demonstrates that, in general, actuator buckling is not reversible.

| Actuator Number | Actuator Stiffness | Actual Buckling Load | Buckling load for first mode | Buckling load for second mode | Percent error for first mode | Percent error for second mode |
|-----------------|--------------------|----------------------|------------------------------|-------------------------------|------------------------------|-------------------------------|
| 1 | 0.012 | 1.92 | 1.43 | 5.70 | 34.68 | -66.33 |
| 2 | 0.011 | 1.62 | 1.09 | 4.36 | 48.65 | -62.84 |
| 3 | 0.006 | 1.24 | 0.70 | 2.79 | 77.87 | -55.53 |
| 4 | 0.015 | 2.54 | 1.54 | 6.17 | 64.73 | -58.82 |
| 5 | 0.013 | 1.80 | 1.44 | 5.76 | 24.99 | -68.75 |
| 6 | 0.008 | 0.80 | 0.84 | 3.36 | -4.83 | -76.21 |
| 7 | 0.012 | 1.60 | 1.57 | 6.27 | 2.00 | -74.50 |
| 8 | 0.013 | 3.10 | 1.96 | 7.83 | 58.36 | -60.41 |
| 9 | 0.014 | 2.70 | 1.78 | 7.12 | 51.60 | -62.10 |
| 10 | 0.007 | 1.10 | 0.66 | 2.66 | 65.60 | -58.60 |

Table 3-1: Actuator buckling results and Euler buckling calculations

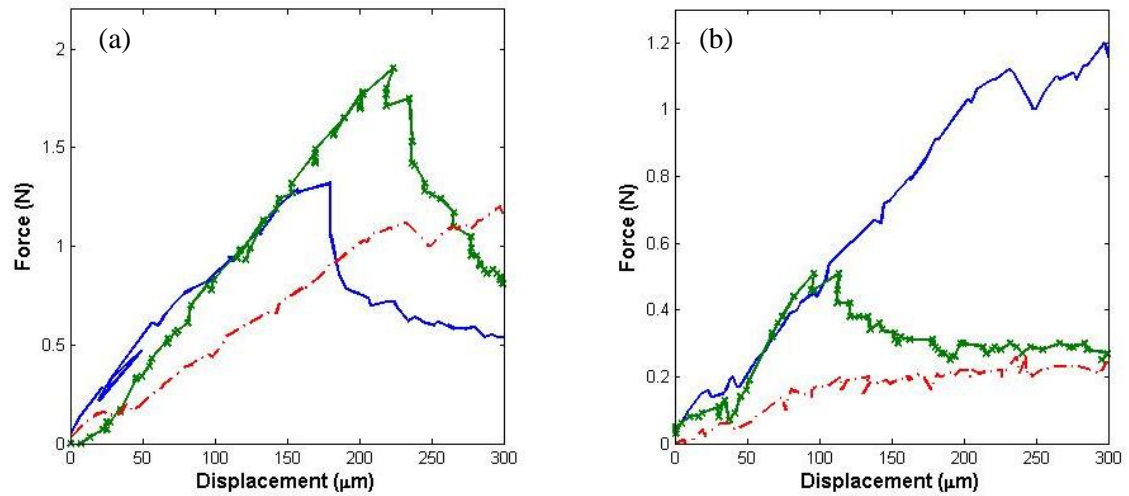


Figure 3-6: (a) Buckling results for three sample actuators. (b) Three experimental runs with the same actuator indicating the first run (blue solid line), the second run (green line with crosses) and the third run (red dot-dash line).

The first buckling mode was typically found to underestimate the buckling load while the second buckling mode was always greater than the experimental value. Figure 3-7 shows a comparison of real buckling load for each actuator, and calculated buckling loads based on actuator stiffness with average values used for actuator diameter, film thickness and initial length. It can be seen that the first and second modes of actuator buckling provide a reasonably effective envelope for actuator buckling results. Linear regression was performed on the experimental buckling results and a best fit experimental effective length of roughly 81 percent of the original length. This indicates that constraining the middle of the actuators effectively increases the buckling load; however, the critical load is substantially less than the theoretical value predicted by the second buckling mode.

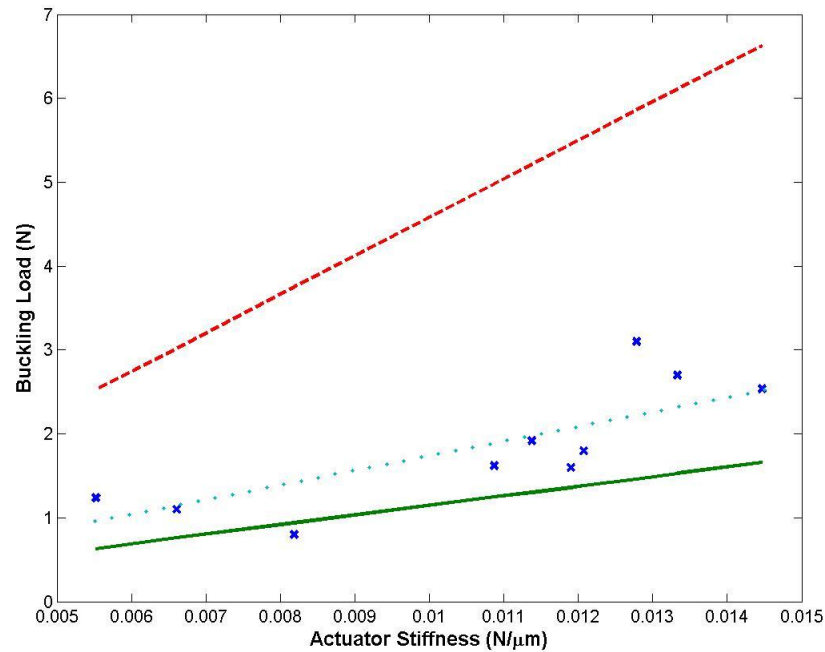


Figure 3-7: Actual and predicted buckling loads plotted against actuator stiffness. Experimental results (blue x), calculated load for first buckling mode (solid green line), calculated load for second buckling mode (red dashed line), and calculated load for best fit model (light blue dotted line)

These buckling results also provide evidence for the behavior observed in previous studies [7], [8]. Data from blocked force testing revealed that the maximum force achieved before a substantial reduction in force occurred was approximately 1.56 Newtons, which is comparable to the 1.84 Newtons found in this study. Additionally, it was previously found that these actuators could support .95 Newtons after buckling, which is relatively close to the average force of .91 Newtons found above. Previous testing was conducted with actuators supported in a continuous tube which was sized to fit closely with the actuators. The similarity of buckling results from these different tests provides further credibility to the theory that supporting the entire length of the actuator does not significantly increase the critical load.

Chapter 4 Handheld Braille Display

Much of the research on implementing electroactive polymers in Braille displays has focused on using the reduced actuator volume to increase the number of characters that can be displayed on a similarly sized device. This study investigated how to implement a refreshable Braille display in a small device. While several sources present concepts for mobile devices with physical outputs for the blind, there are no prominent products on the market today that provide a single solution for mobile computing with a Braille display [12], [13]. Three primary objectives guided the design of this device. First, the overall size should be small compared to existing refreshable Braille devices. Second, the device should receive all of its power and communicate using a standard USB port. Lastly, the device should be easy to fabricate, assemble and repair.

The motivation for developing a handheld Braille display was to provide the blind with an option for receiving small amounts of information from a mobile device without relying on text-to-speech interfaces. Future developments would likely call for three or four characters to improve the functionality of this device; however, even a few characters could present a variety of useful information if users could cycle through a word or phrase. For example, at a grocery store, a display could be used to print the names of scanned products, or it could also be used to scan bills, and print their value. It could also be used as a way to display the time, or as a way to output names in a contact list. It is likely that many other uses for this technology would be found once it was in production.

Mechanical Design

Decreasing the overall device volume was the largest challenge encountered during the design process. In order to reduce device size, the circuits needed to be designed concurrently

with the mechanical components. Figure 4-1 shows the completed handheld Braille display and, and also highlights some of the significant design features. The final assembly was 140 millimeters long, 50 millimeters wide and 18 millimeters high. It was also lightweight with a final mass of less than 90 grams.

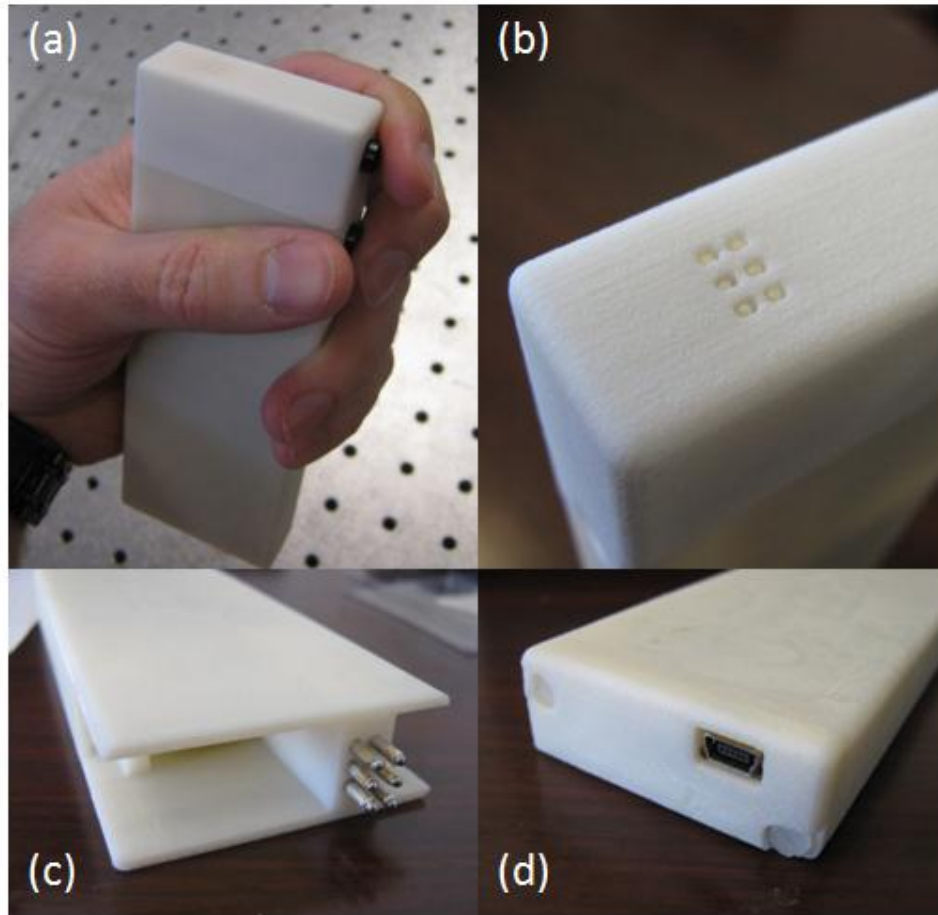


Figure 4-1: Images illustrating the mechanical features of the Braille cell: (a) completed Braille cell, (b) close-up of Braille pins, (c) actuators shown in device, (d) USB computer connection.

All of the major internal components were designed to slide together without permanent connections for quick assembly and disassembly. For example, the main PCB in this unit was held in place by slots in the side of the device and connected to the top of the device and the actuators through removable headers. A PCB was fabricated with surface mount pads that made

the connection to the lower end of the actuators. With the top cap removed, actuators could be quickly removed and installed. The main body of this unit and the lower cap were manufactured with fused deposition modeling, which allowed for quick design changes and would make production of additional prototypes more convenient. The top cap required greater precision, and was made by a selective laser sintering process. Safety was an important concern in the mechanical design, as high voltages are used in this device. To help prevent users from coming in contact with the high voltage supply, all high voltage connections from the main PCB are broken when the top cap is removed. In a final device, further precautions would need to be taken to prevent users from dangers such as electrocution from water entering the device.

Electrical Design

Two main criteria drove the design of the actuator electrical connections: these connections must be temporary in order to facilitate easy actuator replacement, and one connection must maintain contact as the actuators extended and retracted. Also, the moving electrode needed to be compliant enough to not significantly restrict the motion of the actuator. A unique solution to this problem was found by using a material known as conductive fabric. This material (MedTex P-180) is made out of silver plated nylon, and was found to make a reliable electrical connection. This material is also relatively inexpensive with a 12 inch by 13 inch sheet costing approximately 20 dollars. Each character uses less than one square inch, so one sheet would provide enough material to make several displays. In order to ensure that the material was always in contact with the actuators, actuators initially protruded into the conductive fabric by a distance of 1 millimeter. This material was also effective, because it provided enough force to keep the actuators in contact with the lower electrode without significantly constraining actuator movement. Using the force and strain testing device described in chapter 3, it was found that

displacing the fabric in a test cell by 2 millimeters required roughly .1 Newtons of force. Figure 4-2a shows the conductive fabric electrode in a test Braille cell. An issue that arises when using the conductive fabric is the tendency for one extending actuator to raise neighboring pins. To fix this problem, supports were placed in the top of the cell, which help keep the electrode flat. This feature can be seen in figure 4-2b.

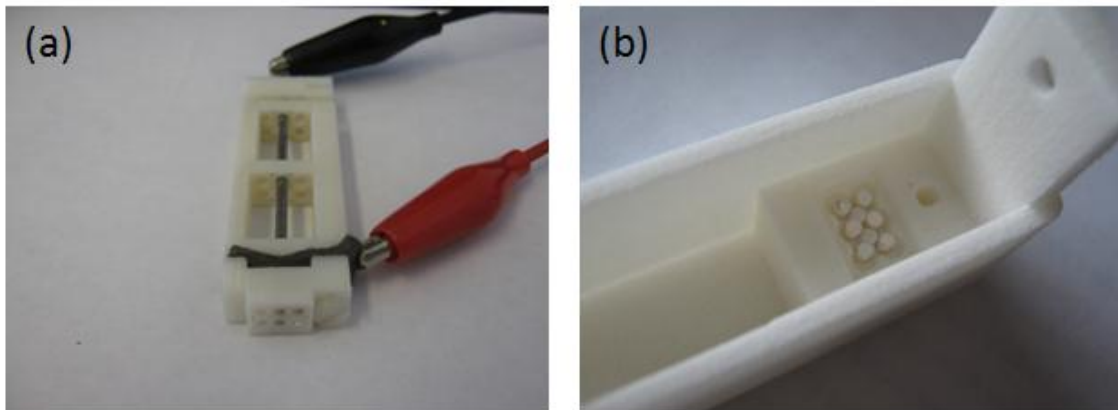


Figure 4-2: (a) Conductive fabric being tested in an experimental Braille cell. (b) Raised features are used to keep the conductive fabric under one pin from interacting with its neighbors.

The electronics in this display needed to be able to supply high voltages to the actuators, and fit in the space available. An Emco A05P-5 amplifier was selected because it met the power requirements at a reasonably low cost. Printed circuit boards were created to reduce device size, increase reliability, and allow multiple prototypes to be quickly constructed. An Arduino microcontroller was selected to handle computer communication tasks and actuator activation. The main device circuit board can be seen in figure 4-3.

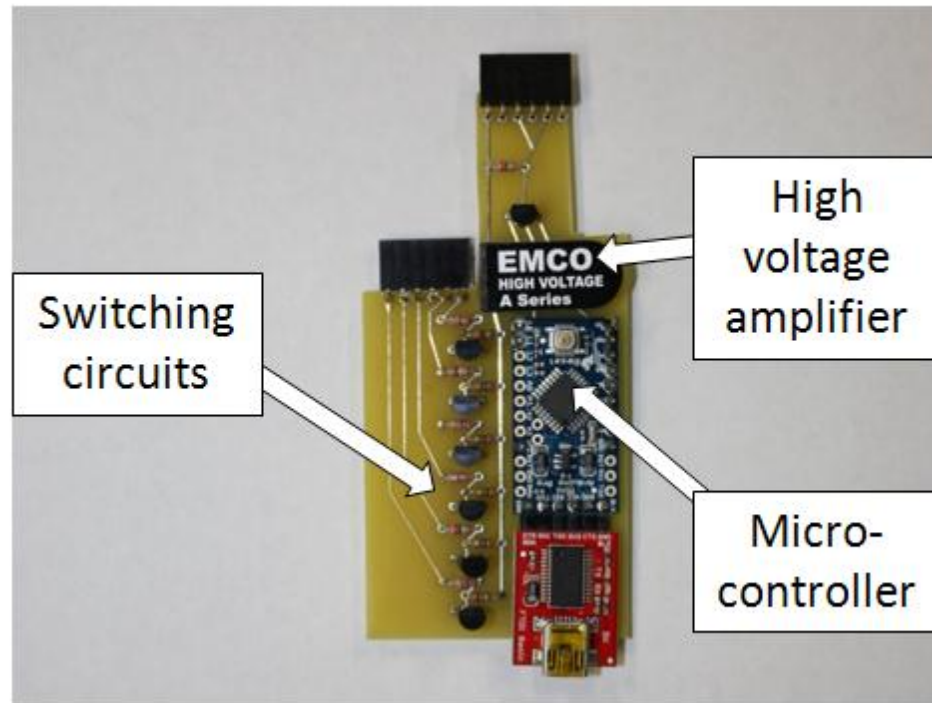


Figure 4-3: Main PCB of the handheld Braille display with part descriptions

Charging and discharging the actuators was accomplished with the circuit shown in figure 4-4. These actuators have been observed to fail at higher currents, so a resistor was placed in series with the actuator to limit current. When the N-channel MOSFET transistor was activated by a 5 Volt signal from the microcontroller, the charging circuit was connected to ground and the actuator extended. A resistor was also needed in this circuit to discharge the actuator when the transistor was deactivated. The resistors and actuator capacitance formed a first-order filter which limits the charging and discharging time. Using typical values for actuator capacitance, it was found that the actuator will have a charging time constant of .5 seconds, and a discharging time constant of 1.25 seconds. While the charge and discharge times currently achievable with this device are slow for a Braille display, as these actuators develop further, response times could potentially be improved. Also, a second transistor could replace the parallel resistor in this circuit,

which would eliminate a majority of the steady-state current when an actuator is active, and would decrease discharging time.

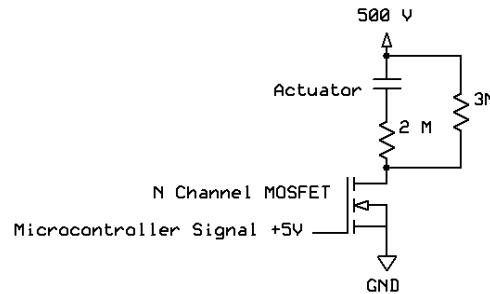


Figure 4-4: Actuator charging and discharging circuit schematic

Embedded Software

The microcontroller software was responsible for communicating with the user's computer and actuating the appropriate pins to produce the desired Braille character. When a string of characters arrived over the USB cable, the software replaced the previous message in memory with the new message. By default, the first character of the new message was displayed. Every lowercase character had an associate byte in the program that indicated which pins needed to be actuated to produce a given character. The software used the value of each bit within a characters byte to either pull a digital output line to 5 Volts, or ground. The software also communicated the currently displayed character to the host computer. Additional information about the software used can be found in appendix D.

User Interface and Digital Communication

A user interface was created with software called Processing to send messages to the handheld display, and receive feedback about the operation of the device. This interface allows a

user to enter messages, which are sent over the USB cable to the handheld device. The user can then cycle through the message with the two buttons on the side of the device. As the user cycles through the message, the character displayed on the device and the computer screen will change simultaneously. Active Braille pins are indicated in the GUI so that proper device performance can be confirmed. A screenshot of the interface can be seen in Figure 4-5.

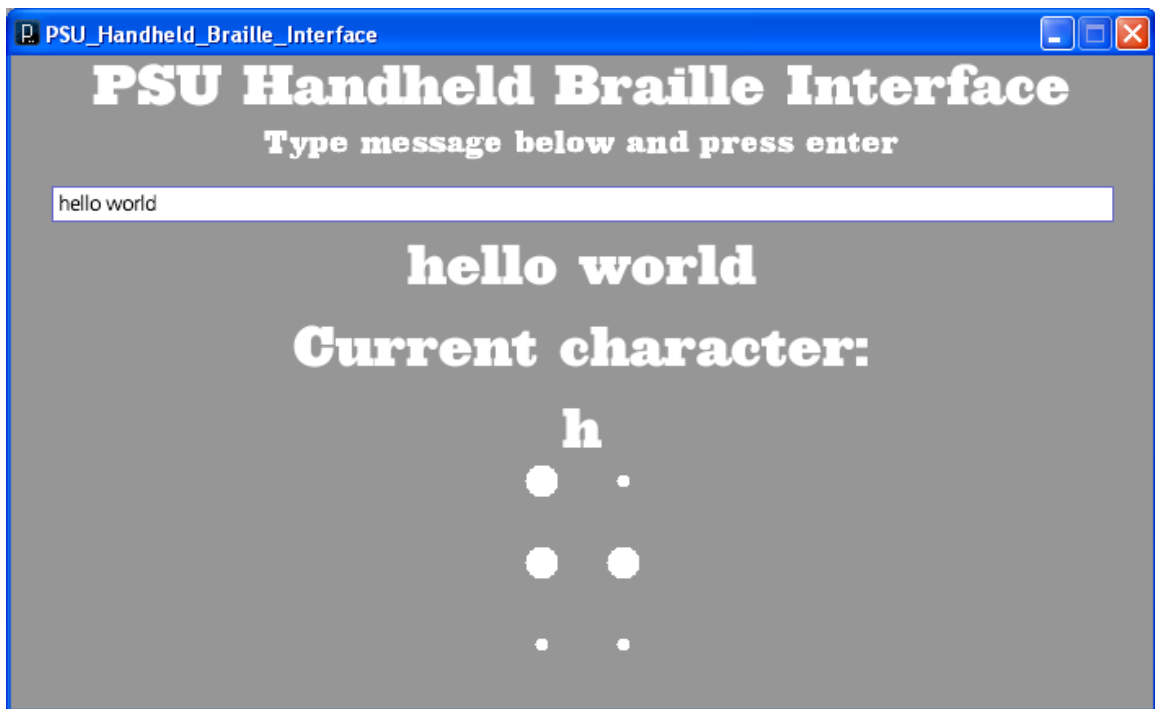


Figure 4-5: Computer interface for handheld Braille display.

Testing

Correct operation of the PCBs was verified before computer control was attempted. With no actuators connected, each of the switching circuits was individually activated to ensure proper operation. Then each circuit was activated sequentially until all six circuits were active. Using an unregulated voltage supply presented a unique challenge in this design, because the output

voltage depends on the output current. At steady state, the resistor-capacitor charging circuit will draw very little current; however, the parallel resistor seen in figure 4-4 will always draw some current when the circuit is active. Figure 4-6 shows data for the output voltage and input current of this circuit. Future circuit designs could utilize a regulated power supply, or incorporate a switching circuit that eliminates the need for an additional resistor across the ends of the actuator. It was also confirmed that pulse width modulation (PWM) can be effectively used to reduce the voltage across the actuator, which will be discussed in greater detail below.

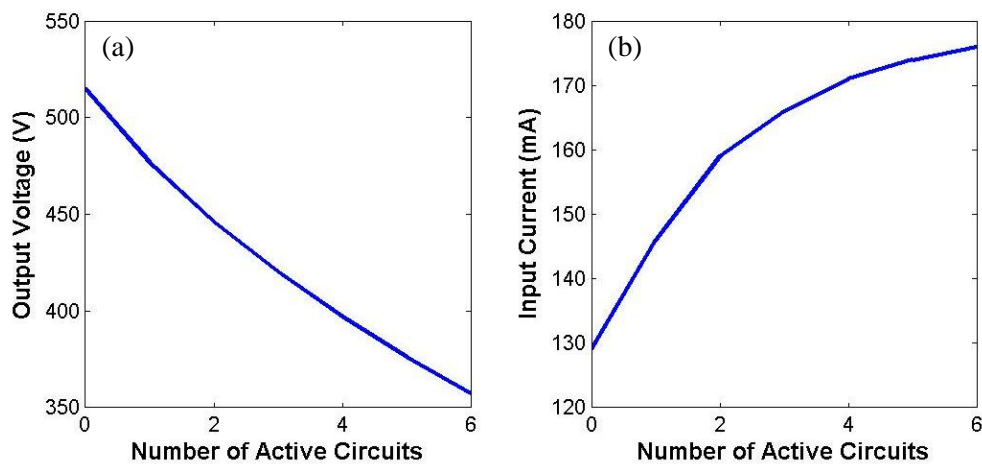


Figure 4-6: Plot (a) shows that the output voltage decreases in response to activating additional charging circuits. Plot (b) shows the relationship between input current and the number of active charging circuits.

Displacement tests were performed on the final device to confirm correct operation. A laser vibrometer was used to precisely measure the displacement of a pin. The experimental setup can be seen in figure 4-7. The test pin was switched on and off every ten seconds, and data were logged using LabVIEW. Displacements of up to .45 millimeters were found, which is slightly less than the Braille requirement of .5 millimeters; however, actuators with a 15 percent longer active area are being developed to address this issue, and it may be possible to use a higher voltage supply to enable greater displacements.



Figure 4-7: Braille pin displacement validation using a laser vibrometer.

Pulse width modulation (PWM) was investigated as a method to control actuator voltage. A sample actuator was tested at different PWM frequencies and duty cycles with no observed detrimental effects on the actuator. As expected, when the duty cycle is reduced, the final displacement is also reduced. Also, when the frequency of the PWM signal is reduced larger oscillations appear in the displacement. These effects can be seen in figure 4-8. While there was some concern that using PWM to control these actuators would cause them to fail sooner, a test actuator was repeatedly operated with PWM with no observed detrimental effects. Previous studies reported a cycle life for similar Braille actuators of roughly 1000 cycles [7], so future testing should confirm that PWM actuation does not negatively affect the life of these actuators.

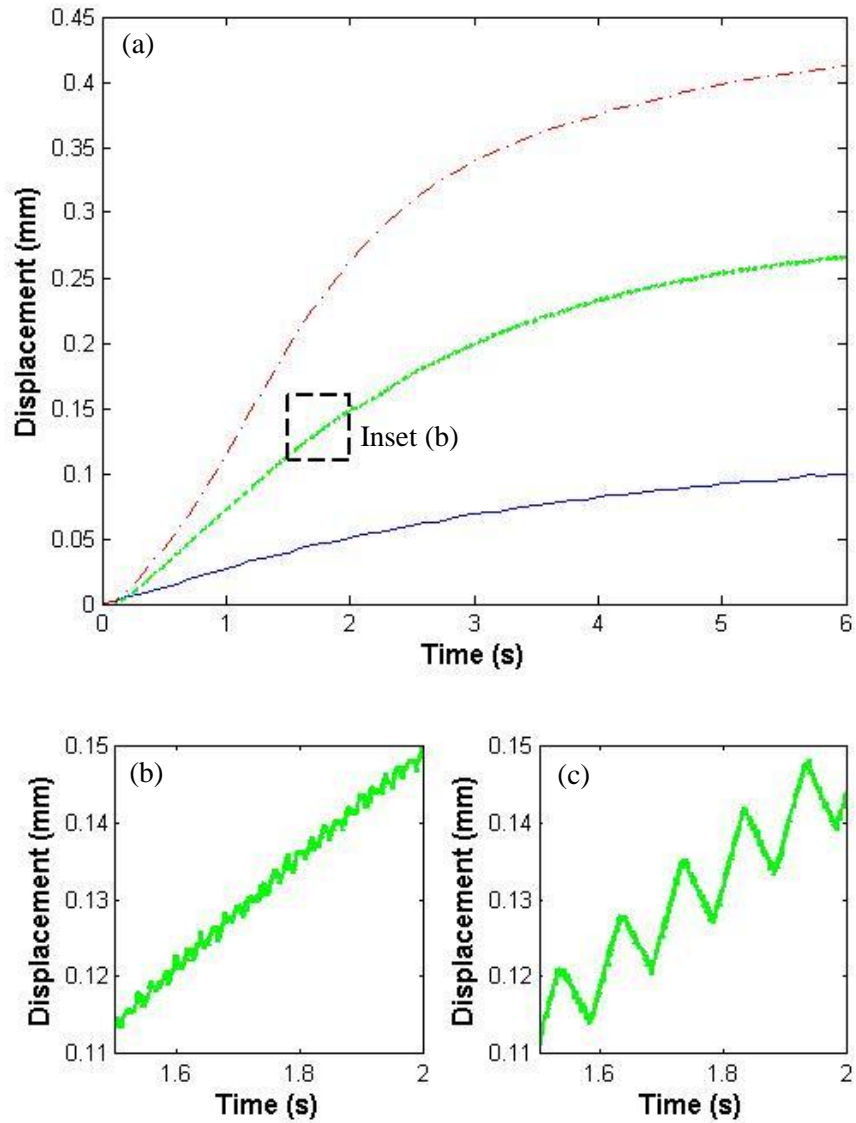


Figure 4-8: Plot (a) shows actuator displacement for three different duty cycles at 50 Hz PWM frequency. The blue solid line is 25 percent duty, the green dashed line is 50 percent and the red dot-dashed line is 100 percent. Plot (b) shows the displacement profile of an actuator with a 50 percent duty cycle and 50 Hz switching frequency. Plot (c) shows the displacement profile of an actuator with a 50 percent duty cycle and a 10 Hz switching frequency. It can be seen that increasing the switching frequency significantly reduces displacement fluctuations.

The device was found to perform as expected. Messages could be entered into the user interface, and the correct circuits were activated for each character. Figure 4-9 shows the final unit displaying a Braille character. In order to account for variability in actuator length, short copper shims were created. These shims were ground to the correct length, and placed between the actuator and the conductive fabric. While this was found to work acceptably, this step would be unnecessary if future actuators can be made to tighter length tolerances. The device also proved to be reliable during testing. Tests were performed over the course of several months with no device failures.

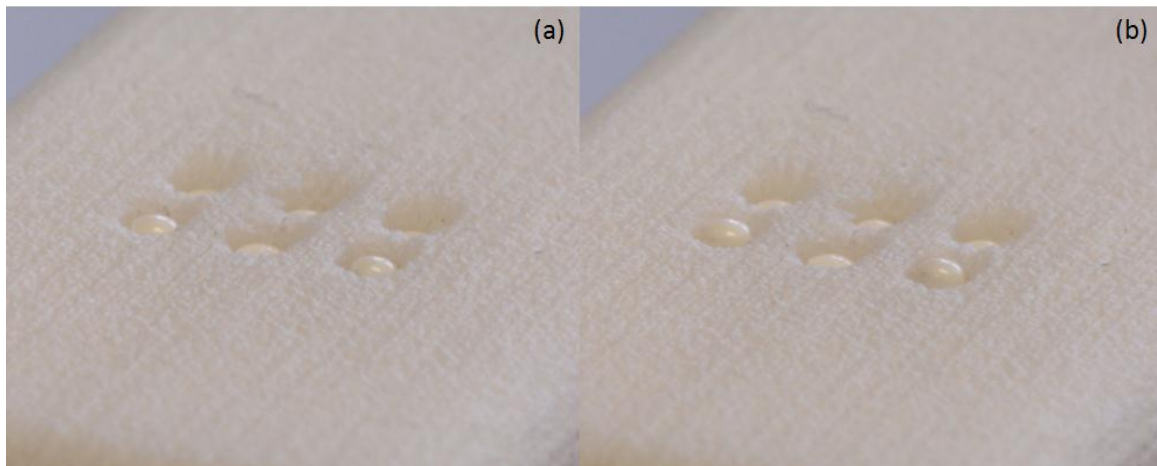


Figure 4-9: Image (a) shows the Braille cell without any actuation voltage and (b) shows the cell displaying the letter “k”, which corresponds to pins 1 and 3 raised.

Chapter 5 Conclusions and Future Work

Mechatronic solutions were developed for the fabrication, testing and implementation of rolled, core-free P(VDF-CFE-TrFE) actuators. It has been shown that a relatively simple machine can effectively and conveniently roll actuators. A tabletop force and strain testing station was successfully built and tested. Stiffness values obtained with this device were used with an Euler buckling model to predict actuator critical buckling loads. These actuators were found to have an average buckling force of 1.84 Newtons when supported at the end and in the middle, which greatly exceeds the Braille standard of .5 Newtons. Lastly, a novel handheld, computer-controlled Braille cell was developed and demonstrated.

The most important feature of all these devices was their simplicity. These tools were designed to have their function be obvious so that the user could focus on the task they are performing. Also, all of these devices were primarily constructed through rapid prototyping methods, which would allow any future changes to be made quickly and easily. These devices were also designed to be flexible so that future requirement changes do not require a major redesign. Lastly, these devices were designed to be inexpensive. Approximate material costs for these devices were as follows: 100 dollars for the actuator rolling machine, 350 dollars for the force and strain testing machine, and 300 dollars for the handheld Braille display (excluding actuators).

Future research could examine how to expand on the handheld Braille display concept. If 3 or 4 cells could be incorporated in to one device, it would be more practical to provide users with small amounts of information quickly. While this prototype Braille display was controlled by a computer, methods of implementing this type of display into a cell phone or a smart phone could greatly benefit the blind community.

References

- [1] Y. Bar-Cohen, "Electroactive polymers for refreshable Braille displays," *SPIE Newsroom*, 10.1117/ 2.1200909.1738, 209.
- [2] R. Kornbluh et al., "Application of Dielectric Elastomer EAP Actuators," *Electroactive Polymer [EAP] Actuators as Artificial Muscles*, pp. 573-573, 2004.
- [3] H. R. Choi et al., "A Braille Display System for the Visually Disabled Using a Polymer Based Soft Actuator," *Biomedical Applications of Electroactive Polymer Actuators*, pp. 427-442, 2009.
- [4] Y. Kato, "Sheet-Type Braille Displays by Integrating Organic Field-Effect Transistors and Polymeric Actuators," *IEEE Transactions on Electronic Devices*, vol. 54, 2007.
- [5] G. M. Spinks and G. G. Wallace, "Actuated Pins for Braille Displays," *Biomedical Applications of Electroactive Polymer Actuators*, pp. 265-277, 2009.
- [6] F. Bauer, E Fousson, Q. M. Zhang, "Recent Advances in highly Electrostrictive P(VDF-TrFE-CFE) Terpolymers," *IEEE Transactions on Electronic Devices*, vol. 13, 2006.
- [7] P. Diglio, "Design, Development and Manufacturing of Rolled Cylindrical PVDF Actuators for Braille Displays," Master's Thesis, The Pennsylvania State University, 2010.
- [8] T. Levard, "Core-free rolled actuators for Braille Displays using eletrostrictive P(VDF-TrFE-CFE) Terpolymer," Master's Thesis, The Pennsylvania State University, 2011.
- [9] D. Norman, "The Design of Everyday Things," 1988.
- [10] "Braille Cell Dimensions" tiresias.org [online], Nov. 20, 2009, [cited Jan. 31, 2012], http://www.tiresias.org/research/reports/braille_cell.htm.
- [11] J. M. Gere, S. P. Timoshenko, "Mechanics of Materials," Third Edition, p. 590 1990.
- [12] S. Levenstein, "Braille Cell Phone Gives Good Vibration to Blind & Deaf Callers" inventorspot.com [online], [cited Feb. 10, 2012], <http://inventorspot.com/articles/>

braille_cell_phone_gives_good_vibrations_blind_deaf_callers_12440.

- [13] M. Romeo et. al., “BrailleTouch, Designing a Mobile, Eyes-Free Soft Keyboard,”
MobileHCI, 2011.

Appendix A: Force and Strain Testing Machine LabVIEW Interface

The LabVIEW interface for the force and strain testing machine can be seen in figure A-1. There are three main modes of control for this device: manual control, displacement control and force control. Each mode is activated by clicking the button below the appropriate label. If more than one button is activated, the manual control mode takes precedence. The plot labeled “Force” shows the force history of the system. Note that the time axis of this plot is in data points, not seconds. The plot labeled “Force and Displacement” shows the history of force and displacement readings. The button labeled “Clear Data” removes all of the prior recorded data. Data can be exported to Microsoft Excel or the system clipboard from LabVIEW. Measured displacement is shown in the box labeled “Displacement” while the commanded displacement is shown in the box labeled “Command”. The current force reading is shown in the box labeled “Force”.

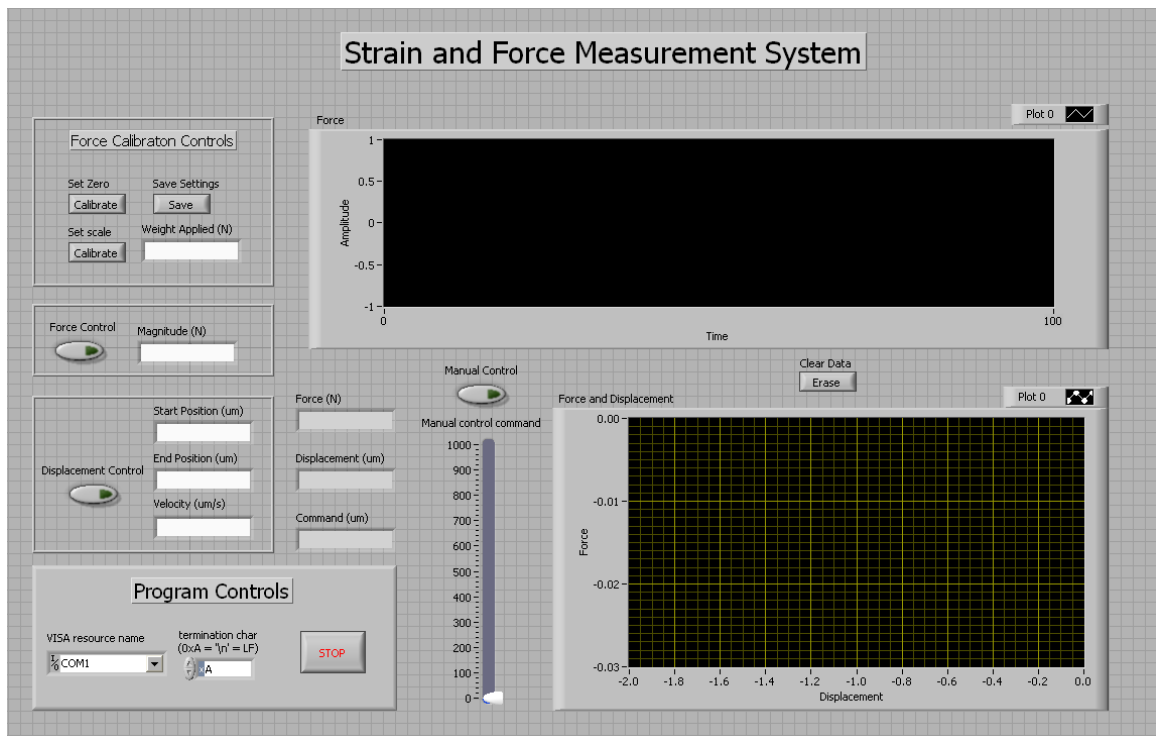


Figure A-1: Force and strain testing machine user interface.

In manual control mode, the slide labeled “Manual Control Command” controls the displacement of the lower vice. The numbers on the scale correspond to micrometers of displacement.

In displacement control mode the user provides a starting position and a final position in micrometers. The user also provides a desired velocity in micrometers per second. Once the toggle switch for the displacement control is activated the lower vice will first move to the starting position then move towards the end position at approximately the rate specified by the velocity. Once the end position is reached, the vice will stop.

In force control mode, the user provides a desired force and the machine will move the lower vice to attempt to maintain that force. The machine slowly moves the lower vice up if the force is lower than expected and down if the force is higher than expected.

Appendix B: Force and Strain Testing Machine Embedded Software

The code listed here was loaded onto an Arduino Uno microcontroller board to handle all of the sensing and control tasks associated with the force and strain testing machine. The position of the linear actuator is controlled by a modified PID algorithm with a tolerance around the set point to prevent oscillations. Communication with LabVIEW is handled through serial strings. A string of characters is sent from LabVIEW and parsed in the Arduino code to exchange data and set all relevant machine parameters.

```

/*
Works with LabVIEW VI Strain_and_Force_Measurement_1_30_12
VI originally implemented with LabVIEW 2010
This code allows the Arduino to communicate with LabVIEW and control an experimental force
and strain testing station.

Supply voltage is tuned for 12V. Other voltage supplies may require different control gains

Last edit: Michael Robinson 1/30/12
*/

#include <TimedAction.h> //Used to control timing in displacement control mode
#include <EEPROM.h> //Used to store calibration settings for force transducer

TimedAction displacementUpdate = TimedAction(50,inc);

//Motor control and feedback variables
int commandPin=0;
int feedbackPin=1;
int strainGauge = 2;
int m1PWM=9;
int m1A=11;
int m1B=10;

int actuatorZero=800;//Linear actuator value for zero displacement

float command=actuatorZero; //Linear actuator set point
float feedback=0; //Current linear actuator position

float pGain=2;
float iGain=2;
float dGain=-120;

```

```
float derivative=0;
float integral=0;

float pastFeedback=0; // Past position

float omega=0;
float diff=0;//error value = set point - current position
int actuatorBias=50;//Duty cycle required to overcome actuator friction
int tol=3;//Linear actuator will not move if current position is within tolerance of set point
int tol2=10;//Linear actuator will move with reduced voltage below this tolerance
int maxPower=100;//Variable used to store current maximum duty cycle
int maxPower1=70;//Maximum duty cycle when current position is around set point
int maxPower2=150;//Maximum duty cycle otherwise
int posMax=900;
int posMin=150;
//for 9V power maxPower1=100, maxPower2=200, actuatorBias=80

//Displacement Control Variables
int displacementStart=700;
int displacementEnd=500;
int displacementRate=100;
float umPerCount=2.25;//Micrometers of lower vice displacement per actuator potentiometer
count

//Force control variables
float zero=380;
float pastRef=380;
float scale=30;
float forceRef=1;
float force=0;
float pastADC=0;
float forceSetPoint=0;
float fControlGain=.1;
float fControlTol=.05;
float forceDifference=0;
int forceMax=6;

//Program control variables
int forceControl=0;
int manualControl=0;
int displacementControl=0;

//Serial Communication Variables
char buffer[20];
```

```

void setup() {
  Serial.begin(9600);
  pinMode(m1A,OUTPUT);
  pinMode(m1B,OUTPUT);
  scale=EEPROM.read(0);
  zero=EEPROM.read(1)*4;
}

void loop() {
  communicate();
  readForce();

  if(displacementControl)
  {
    displacementUpdate.check();
  }

  Serial.print(force);
  Serial.print(" ");
  Serial.print(float(actuatorZero-feedback)*umPerCount);//Print Displacement
  Serial.print(" ");
  Serial.println(float(actuatorZero-command)*umPerCount);

  for(int jj=0;jj<80;jj++)
  {
    control();
    if(forceControl)
    {
      fControl();
    }
  }
}

//Reads the analog feedback from the actuator potentiometer
//on the pin specified by the variable feedbackPin
//Controls the position of the linear actuator using a rough equivalent of PI control
//A bias voltage is applied to the actuator to overcome the initial effects of friction
//A tolerance is used to prevent actuator oscillations
void control()
{
  feedback=analogRead(feedbackPin);
  derivative=(pastFeedback-feedback)*dGain;
  pastFeedback=feedback;
  diff=command-feedback;
  if(abs(diff)<tol2)maxPower=maxPower1;
  else maxPower=maxPower2;
}

```

```

//Pseudo integrator
if(derivative==0&&abs(diff)>tol)
{
  integral++;
}
else integral=0;

//Control direction of motor
if(displacementControl&&diff<0)
{
  digitalWrite(m1A,HIGH);
  digitalWrite(m1B,LOW);
  if(feedback>posMin)omega=abs(diff)*2*pGain+actuatorBias+derivative+integral*iGain+2;
  else omega=0;
}
else if(diff<(-1*tol))
{
  digitalWrite(m1A,HIGH);
  digitalWrite(m1B,LOW);
  if(feedback>posMin)omega=abs(diff)*pGain+actuatorBias+derivative+integral*iGain+2;
  else omega=0;
}
else if(diff>tol)
{
  //move actuator up
  digitalWrite(m1A,LOW);
  digitalWrite(m1B,HIGH);
  if(feedback<posMax)omega=abs(diff)*pGain+actuatorBias-derivative+integral*iGain-50;
  else omega=0;
}

else omega=0;
if(omega>maxPower&displacementControl==0) omega=maxPower;
if(omega>maxPower&displacementControl==1&diff>0) omega=maxPower;

analogWrite(m1PWM,omega);
delayMicroseconds(500);
}

```

```

//Handles all communication with LabVIEW.
//A string is passed over the serial line and different characters are used as delimiters
//the function communicate parses this string to set variable values
void communicate()
{
    int length=20;

    if(Serial.available(>0)
    {
        int ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        while(Serial.available(>0)&&ii<length)
        {
            buffer[ii]=Serial.read();
            switch(buffer[ii])
            {
                case 'p':
                    command=atoi(buffer);
                    ii=0;
                    for(int jj=0;jj<length;jj++) buffer[jj]=0;
                    break;
                    //Displacement control variables
                case 'd':
                    displacementStart=atoi(buffer);
                    displacementStart=actuatorZero-float(displacementStart)/umPerCount;
                    if(displacementControl==0)
                    {
                        displacementControl=1;
                        command=displacementStart;
                    }
                    ii=0;
                    for(int jj=0;jj<length;jj++) buffer[jj]=0;
                    break;

                case 'e':
                    displacementEnd=atoi(buffer);
                    displacementEnd=actuatorZero-float(displacementEnd)/umPerCount;
                    ii=0;
                    for(int jj=0;jj<length;jj++) buffer[jj]=0;
                    break;

                case 'm':
                    displacementControl=0;
                    forceControl=0;
                    manualControl=1;
                    ii=0;
                    for(int jj=0;jj<length;jj++) buffer[jj]=0;
                    break;
            }
        }
    }
}

```



```
    case 'r':
        displacementRate=atoi(buffer);
        ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        break;

    case 's':
        forceRef=atof(buffer);
        scale=pastRef/forceRef;
        ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        break;

    case 'z':
        zero=pastADC;
        ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        break;

    case 'f':
        forceSetPoint=atof(buffer);
        forceControl=1;
        ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        break;

    case 'c':
        EEPROM.write(0, scale);
        EEPROM.write(1, zero/4);
        ii=0;
        for(int jj=0;jj<length;jj++) buffer[jj]=0;
        break;

    default:
        ii++;
        break;
}
delay(1);

}
}
}
```

```

//Advances the platform when in displacement control mode
//Note that displacement rates are approximate
void inc()
{
  float increment=0;
  increment=float(displacementRate)/umPerCount*.1;
  if(displacementEnd<displacementStart)
  {
    if(command>displacementEnd)command=command-increment;
    else command=displacementEnd;
  }

  else
  {
    if(command<displacementEnd)command=command+increment;
    else command=displacementEnd;
  }
}

//Reads the analog voltage on the pin specified by the variable strainGauge.
void readForce()
{
  float adcValue=0;
  int numAverage=10;
  int numClear=30;

  //Use the internal 1.1V reference to improve the accuracy of the strain gauge measurement
  analogReference(INTERNAL);
  //Take a few readings that won't be used to set the new reference voltage
  for(int ii=0; ii<numClear; ii++)
  {
    adcValue = adcValue+analogRead(strainGauge);
  }
  adcValue=0;

  //Read ADC value to be used in measurement
  for(int ii=0; ii<numAverage; ii++)
  {
    adcValue = adcValue+analogRead(strainGauge);
  }
  adcValue=adcValue/numAverage;
  pastRef=(zero-adcValue);
  pastADC=adcValue;
  force=(zero-adcValue)/scale;

  //Reset the analog reference to 5V and take a few readings
  analogReference(DEFAULT);

```

```
for(int ii=0; ii<numClear; ii++)  
{  
  adcValue = adcValue+analogRead(feedback);  
}  
}
```

```
//Force control function  
//Actuates lower vice to maintain a force supplied by LabVIEW  
void fControl()  
{  
  
  forceDifference=forceSetPoint-force;  
  
  if(abs(forceDifference)>fControlTol&&command<posMax&&command>posMin&&abs(force)<  
  forceMax)  
  {  
    command=command+fControlGain*forceDifference;  
  }  
}
```

Appendix C: Force and Strain Testing Machine Circuit

Circuits were fabricated on an Arduino daughter board, which is also known as a shield. A complete circuit schematic can be seen in figure C-1. An SN754410 motor driver was used to supply inputs to the linear actuator. Direction and pulse width modulation (PWM) voltage commands were provided by digital output pins on the Arduino. The linear actuator feedback potentiometer was read into the Arduino analog input pin 1. Strain gauge conditioning and amplification were performed with an LM 324 op-amp and an AD623 instrumentation op-amp. A potentiometer was used to bias the AD623 output up to roughly .5 volts. This allowed both positive and negative strains to be measured. The Arduino ADC was set to have a range of 0 to 1.1 Volts. A low-pass filter was also placed on the output of the AD623 to provide some signal smoothing.

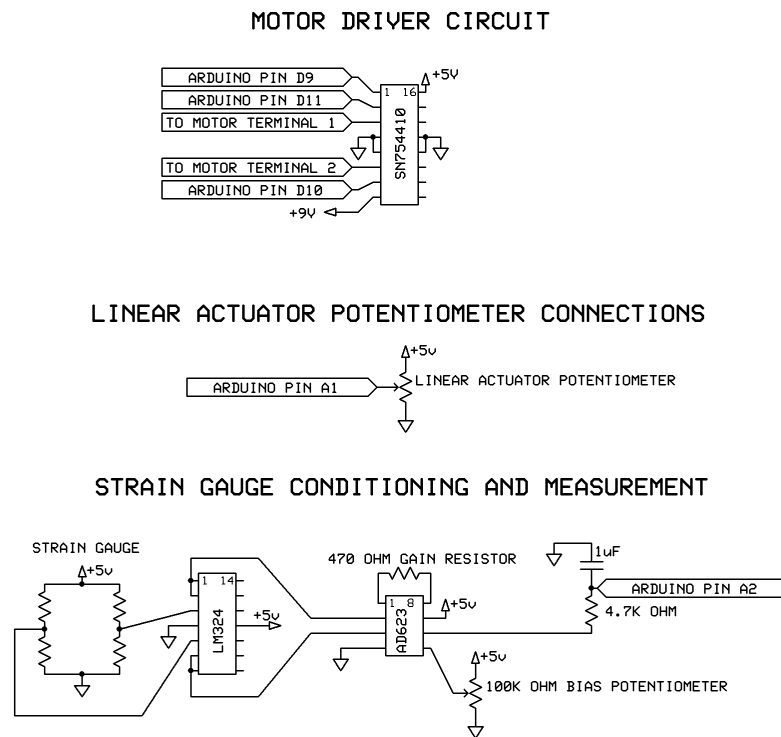


Figure C-1: Circuit schematics for force and strain testing machine

Appendix D: Handheld Braille Cell Embedded Software

The following code was uploaded onto the Arduino microcontroller board to handle computer communication and actuator activation. Pin numbering follows the standard Braille convention seen in figure D-1.

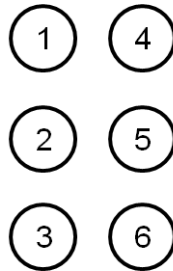


Figure D-1: Standard Braille pin numbering

```

/*
PSU handheld Braille display microcontroller code

This code handles serial communication with a computer and
switching the high voltage signals to the Braille actuators

pin numbering follows Braille standard as seen below:
  p1 p4
  p2 p5
  p3 p6

Last edit: MDR 1/30/12
*/
//Map between Braille pins and digital pins
const int p1=5;
const int p2=4;
const int p3=3;
const int p4=6;
const int p5=7;
const int p6=2;
////////////////////////////////////

//String message=" ";//Holds message sent from computer
char currentCharacter='a';//Character currently shown on display
int charIndex=0;//Location of character within message

char buffer[20];//Buffer of characters to receive from computer

```

```
int length=20;//Length of buffer
byte pinState = B000000;

void setup() {
  // initialize the digital pin as an output.
  Serial.begin(9600);

  buffer[0]=' ';//Set initial character to a space

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);

  pinMode(8, INPUT);
  pinMode(9, INPUT);

  //Activate internal pull-up resistors
  digitalWrite(8, HIGH);
  digitalWrite(9, HIGH);

  //Initially deactivate all actuators
  digitalWrite(p1, LOW);
  digitalWrite(p2, LOW);
  digitalWrite(p3, LOW);
  digitalWrite(p4, LOW);
  digitalWrite(p5, LOW);
  digitalWrite(p6, LOW);

  //Wait for computer communication
  while(Serial.available()==0)
  {
    delay(1);
  }
}

void loop() {
  communicate();
  decode();
  setStates();
  readButtons();
  currentCharacter=buffer[charIndex];
  Serial.print(255,BYTE);
  Serial.print(currentCharacter);
  Serial.println(pinState,BYTE);
}
//Handle all communication with host computer
```

```
void communicate()
{
    if(Serial.available(>0)
    {
        int ii=0;

        for(int jj=0;jj<length;jj++) buffer[jj]=0;//Reset Character Buffer
        charIndex=0; //Reset to display first character of message;
        while(Serial.available(>0&&ii<length)
        {
            buffer[ii]=Serial.read();
            ii++;
            delay(1);
        }
    }
}
//Convert between characters and pin states
void decode()
{
    switch(currentCharacter)
    {
        case 'a':
            pinState=B10000000;
            break;

        case 'b':
            pinState=B11000000;
            break;

        case 'c':
            pinState=B10010000;
            break;

        case 'd':
            pinState=B10011000;
            break;

        case 'e':
            pinState=B10001000;
            break;

        case 'f':
            pinState=B11010000;
            break;

        case 'g':
            pinState=B11011000;
            break;
    }
}
```

```
case 'h':  
pinState=B11001000;  
break;
```

```
case 'i':  
pinState=B01010000;  
break;
```

```
case 'j':  
pinState=B01010000;  
break;
```

```
case 'k':  
pinState=B10100000;  
break;
```

```
case 'l':  
pinState=B11100000;  
break;
```

```
case 'm':  
pinState=B10110000;  
break;
```

```
case 'n':  
pinState=B10111000;  
break;
```

```
case 'o':  
pinState=B10101000;  
break;
```

```
case 'p':  
pinState=B11110000;  
break;
```

```
case 'q':  
pinState=B11111000;  
break;
```

```
case 'r':  
pinState=B11101000;  
break;
```

```
case 's':  
pinState=B01110000;  
break;
```



```
case 't':  
pinState=B01111000;  
break;
```

```
case 'u':  
pinState=B10100100;  
break;
```

```
case 'v':  
pinState=B11100100;  
break;
```

```
case 'w':  
pinState=B01011100;  
break;
```

```
case 'x':  
pinState=B10110100;  
break;
```

```
case 'y':  
pinState=B10111100;  
break;
```

```
case 'z':  
pinState=B10101100;  
break;
```

```
case '1':  
pinState=B10000000;  
break;
```

```
case '2':  
pinState=B01000000;  
break;
```

```
case '3':  
pinState=B00100000;  
break;
```

```
case '4':  
pinState=B00010000;  
break;
```

```
case '5':  
pinState=B00001000;  
break;
```

```
case '6':
```

```

    pinState=B00000100;
    break;

    default:
    pinState=00000000;
    break;

}
}
//Activate the appropriate transistors to form each character
void setStates()
{
    digitalWrite(p1,bitRead(pinState,7));
    digitalWrite(p2,bitRead(pinState,6));
    digitalWrite(p3,bitRead(pinState,5));
    digitalWrite(p4,bitRead(pinState,4));
    digitalWrite(p5,bitRead(pinState,3));
    digitalWrite(p6,bitRead(pinState,2));
}
//Check the button states
void readButtons()
{
    for(int ii=0; ii<10; ii++)
    {
        if(digitalRead(9)==0)
        {
            charIndex++;
            if(byte(buffer[charIndex])==0)
            {
                charIndex--;
            }
            while(digitalRead(9)==0)
            {
                delay(1);
            }
            delay(50);
        }
        if(digitalRead(8)==0)
        {
            if(charIndex>0) charIndex--;
            while(digitalRead(8)==0)
            {
                delay(1);
            }
            delay(50);
        }
        delay(25);
    }
}
}

```

Appendix E: Handheld Braille Cell Circuit

The circuitry that controlled the handheld Braille cell is relatively simple, and with the exception of the high voltage amplifier, consists of commonly available circuit parts. An Arduino microcontroller board was used to control this system; however, there are many other readily available microcontrollers that could perform this function.

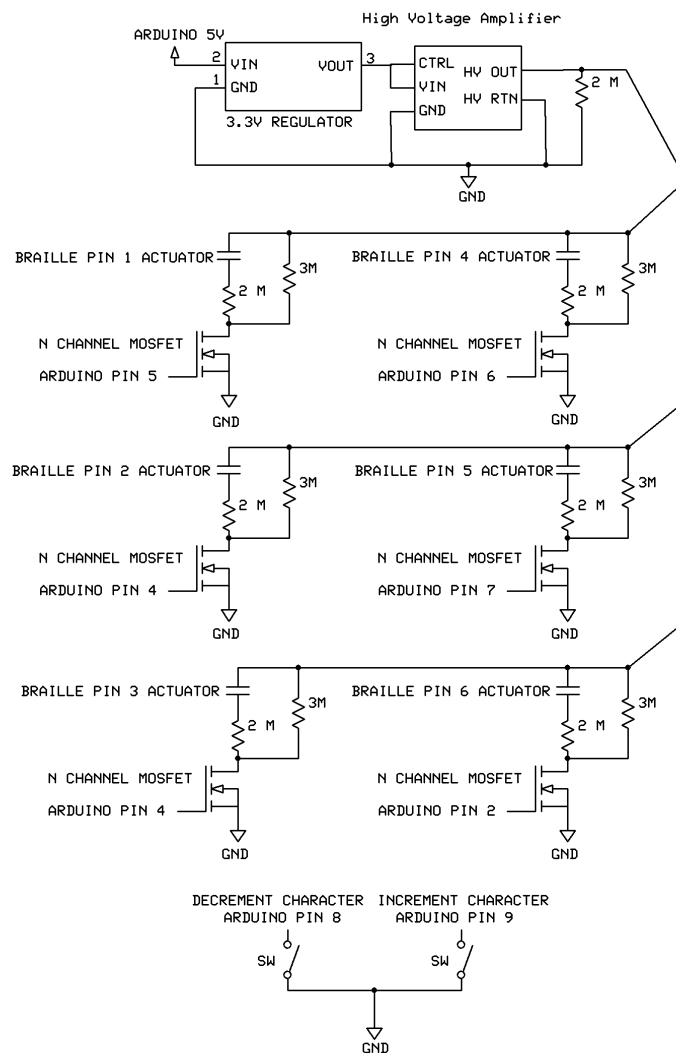


Figure E-1: Handheld Braille display circuit schematic