Title
Mechanical Design of Graphene Nanoribbon Compliant Mechanisms for Electrostatic Discharge

Author
Jones, Talmage

Publication Date
2018

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Mechanical Design of Graphene Nanoribbon Compliant Mechanisms for Electrostatic Discharge

A thesis submitted in partial satisfaction

of the requirements for the degree of Master of Science

in Mechanical Engineering

by

Talmage Harnish Jones

2018

ABSTRACT OF THE THESIS

Mechanical Design of Graphene Nanoribbon Compliant Mechanisms for Electrostatic Discharge

by

Talmage Harnish Jones

Master of Science in Mechanical Engineering

University of California, Los Angeles, 2018

Professor Jonathan Hopkins, Chair

The purpose of this research is to investigate the design of an electrostatic discharge protection device made of single-layer graphene nanoribbons. The device is meant to trigger electrostatic discharge at a target voltage of 1.5V. Other design requirements include the minimization of parasitic capacitance, electrical response time and mechanical response time. The device is designed to discharge static electricity by being pulled to ground through electrostatic forces, then making contact with ground before returning to its original position. Previous designs experienced repeatability issues due to a lack of securing the ribbon and mechanical failure due to high stresses at the boundary conditions. New designs are presented and optimized to maintain a high effective spring constant for the device while reducing stress during electrostatic pull-in. A single-degree-of-freedom model is used in conjunction with the Bernoulli-Euler beam equations and Castigliano's method to guide the design process. Multi-degree-of-freedom and finite element

models are used to validate the predicted pull-in behavior of the new devices and to explore how stresses and reaction forces might affect the reliability. A residual PMMA layer that results from the fabrication process is also incorporated into the finite element model. Molecular dynamics simulations are performed to further explore the behavior of graphene nanoribbons under electrostatic pull-in and to check the accuracy of the finite element approach. The fabrication process is explained and experimental results for the new devices are reported.

The dissertation of Talmage Harnish Jones is approved.

Robert N Candler

Ya-Hong Xie

Jonathan Hopkins, Committee Chair

University of California, Los Angeles

2018

Dedicated to my little Florence

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to acknowledge

… the U.S. Department of Defense's SMART Scholarship program as my source of funding throughout my time as a graduate student.

… Dr. Jonathan Hopkins who helped guide me as I conducted my research.

… the members of the Flexible Research Group at UCLA for their help, especially Mike Porter for introducing me to molecular dynamics.

… Jimmy Ng and Dr. Ya-Hong Xie, who introduced me to this research topic and with whom I collaborated on several papers in producing the presented work.

… my mom and dad, who have always been there to help me succeed and who raised me with a knowledge of the gospel of Jesus Christ, who has been the means of countless blessings in my life.

… my wife Leica, who has shown her constant love and support for me every day, and our little daughter Florence who never fails to brighten my day.

Chapter 1

Introduction

This chapter introduces graphene and its material properties along with electrostatic discharge (ESD) protection devices and previous work done at UCLA to design a graphene ESD device. Chapter 2 describes the process by which new graphene ESD devices were designed and optimized. Chapter 3 reports on an analysis of the new designs using finite-element software. Chapter 4 presents an atomistic study used to assess the accuracy of the finite-element representation of graphene nanoribbons. Chapter 5 describes the fabrication of the new graphene ESD devices and reports the results of device testing.

1.1 Motivation for the Work Presented and Contributions to the Field

The purpose of this research is twofold: (1) to design an electrostatic discharge device made of a graphene nanoribbon and (2) to explore and improve methods for mechanical design of graphene nanoribbon compliant mechanisms.

Previous research in the Materials Science Engineering department at UCLA has attempted to make an ESD device using a chemical vapor deposition and etching fabrication process, described in Chapter 5. This research improves on those designs by applying principles of compliant mechanism design.

This work is expected to offer new contributions to the areas of graphene research and compliant mechanisms in the following forms:

- A graphene ESD device is a novel solution to ESD problems and potentially presents significant advantages over existing solutions.

- In the process of designing such a device, this research explores the effectiveness of various tools in designing such a device. This is seen as a step toward developing a systematic design process for mechanical devices made of graphene nanoribbons. A well-defined design process does not currently exist.
- It provides insight on graphene as a candidate material for compliant mechanisms, especially lamina-emergent mechanisms.
- It models the electrostatic actuation of graphene nanoribbons up to pull-in.

1.2 Background on Graphene

1.2.1 Properties of Graphene

Due to its remarkable electrical and mechanical properties, graphene has been widely researched since its discovery in recent years. In 2004, large graphene crystals were first successfully isolated from bulk graphite material using adhesive tape. Since then, MEMS fabrication processes have been adapted to artificially produce single-layer and multi-layer graphene in single crystalline (SCG) and polycrystalline (PCG) structures[1–4]. In its single-layer form, graphene appears as a two-dimensional array of carbon atoms arranged in a hexagonal pattern, as shown in Figure 1.

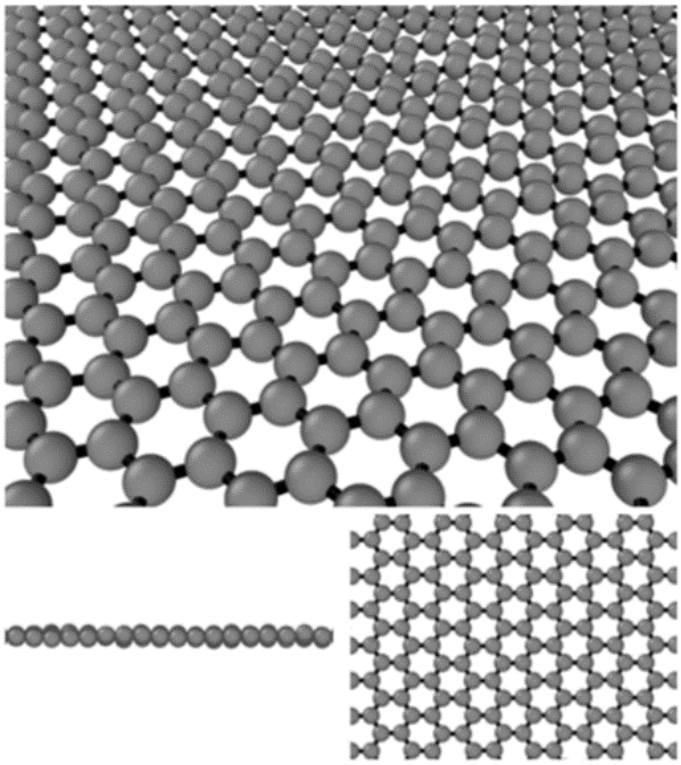

Figure 1: Structure of single-layer graphene as it appears in a molecular dynamics simulation5,6 shown from a perspective view (top), side view (left) and top view (right)

SCG's fracture strength of 130GPa at a strain of 25% and its Young's modulus of 1TPa indicate an exceptionally strong and rigid material. PCG also has a high fracture strength of 46GPa at a 9% strain and a Young's modulus of 600GPa[7–10]. Graphene's atomically thin nature, however, gives it a sub-nanometer thickness of about 3.4Å, allowing for an extremely high flexibility under out-of-plane loads[8]. Additionally, SCG graphene is reported to have an endurance limit of 40GPa under cyclical loading[9]. This combination of high strength and flexibility make graphene a unique material in compliant mechanism design. Graphene's electrical properties make it an ideal candidate for use in MEMS/NEMS devices. Its charge carrier mobility of about 15000cm^2/Vs and doping concentration of about $1\times10^{13}cm^{-2}$ at room temperature allow the material to behave as a great electrical conductor[1]. SCG has an electrical resistance of $2.6\times10^{13}\Omega/m^2$ and can carry a

maximum current of $10A/cm^2$ [9,11]. Use of graphene as a conductor would result in smaller device footprints and a lower parasitic capacitance (about 10 times lower than copper's parasitic capacitance). Graphene's high thermal conductivity of $5\times10^3 W/mK$ (about 13 times as high as copper's) would help prevent device failure due to overheating[2].

1.2.2 Review of Applications for Graphene as a Mechanical Device

Recent mechanical applications for graphene ribbons include electrostatic actuators, resonators, in-plane tension springs to achieve high elasticity and other spring devices analogous to paper kirigami designs[12–18]. However, A reliable and systematic process for the mechanical design of devices made of graphene ribbons has yet to emerge.

This research reports on the design of a novel single-layer graphene nanoribbon device for discharging undesired static electricity from integrated circuits. Electronic devices often experience situations where static electric charges can build up unintentionally. These situations include fabrication, shipping and human interaction. Unless discharged, the static electricity could possibly damage the circuitry to the point where the device can no longer function.

Traditionally, electrostatic discharge (ESD) protection circuits have been created using Zener diodes and grounded gate MOSFETs. These solutions, however, result in a complex circuit design and require a voltage reference to operate. A passively actuated mechanical device is a potentially simpler solution. A flexible conductor can be designed as a mechanical switch that allows undesired charges to flow to ground when a threshold voltage has been crossed. Graphene is the ideal material for such a switch due to its mechanical strength and its ability to conduct a large amount of current before failure[19].

The rest of the chapter gives further explanation of the behavior of such a device and a description of an initial attempt to create such a device. In later chapters, three new designs are presented and analyzed.

1.3 Introduction to Device Functionality

1.3.1 Design Requirements

The objective of this study is to create a single-layer PCG nanoribbon geometry that can be etched using the photolithography and chemical vapor deposition processes used for a previous design[4,19]. Suspended over a doped silicon surface that lies at the bottom of a trench etched from a dielectric layer, the device can fill up with electrostatic charge and then, through electrostatic actuation, be drawn toward the electric ground until making contact to discharge the ribbon. After discharge the ribbon must return to its original position, as shown in Figure 2.

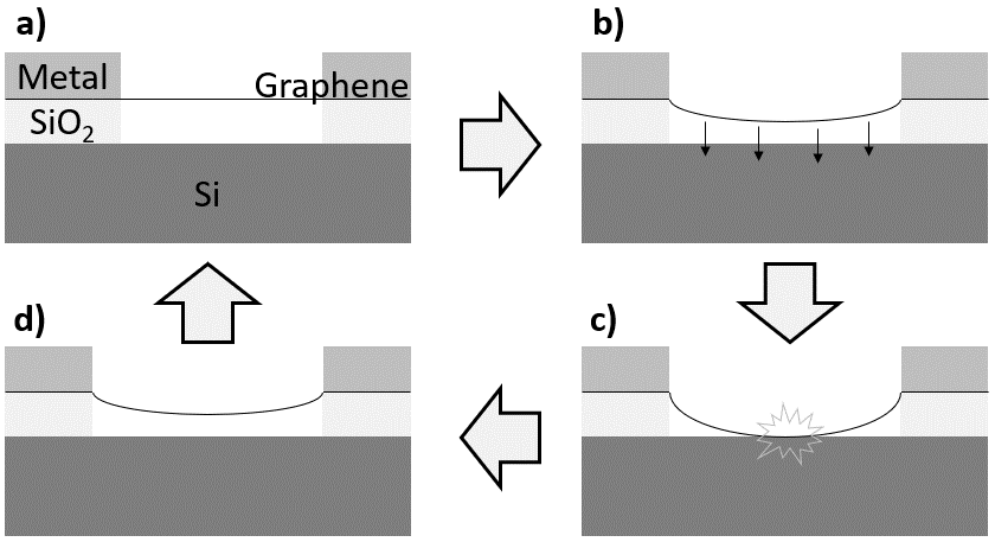

Figure 2: Repeatable function of the graphene ribbon ESD device involving (a) equilibrium position without charge, (b) pull-in by electrostatic forces, (c) contact with electric ground and electrostatic discharge, and (d) release and return to original position

The charging, actuation, discharging and returning must be a repeatable cycle that allows for continued ESD and does not inhibit normal functionality of the integrated circuitry that contains the device.

The devices in this research are meant to eliminate any undesired static charge above 1.5V from the device. The chosen voltage is a safe amount above a 0.8V supply voltage, which could be used to power he most advanced microelectronic technology node as of 2015[20]. Additional design requirements include the minimization of the device's parasitic capacitance, electrical response time and mechanical response time. The parasitic capacitance, a measure of the undesired charges that are stored in the ESD device during normal circuit operation, must be below 150fF. The electrical response time, a measure of the time required for the device to fill with static charge, must be below 1ns. These two electrical requirements are specified by the Industry Council on ESD Target Levels[21]. The mechanical response time, a measure of the time required for the actuation of the device up to discharge, should be on the order of nanoseconds to quickly remove the charge from the circuit.

Dimensions for the design are limited by the fabrication capabilities available to the collaborators in this research. Minimum feature size is 3μm and maximum depth of an etched trench is 1μm.

1.3.2 Pull-in Behavior

The function of the ESD device relies on a phenomenon called electrostatic pull-in. Pull-in of a single-degree-of-freedom (SDOF) system can be illustrated with a simple mechanical mass-spring model, as seen in Figure 3[22]. In the model a mass, m, is suspended by a spring of stiffness, k, above a surface. When a voltage of V_{app} is applied across the gap of distance d, a capacitive

effect is created between the mass and the surface. The result is two forces acting against each other: an electrostatic force that increases with the square of x, and a spring force that increases linearly with x. Damping and van der Waals effects are neglected, and parallel plate capacitor behavior is assumed. The resulting equation of motion is the following:

$$m\ddot{x} + kx = \frac{\varepsilon_0 A V_{app}^2}{2(d-x)^2} \tag{1}$$

where ε_0 is the free space permittivity and A is the surface area of the bottom of the mass experiencing the capacitive effect.

Low applied voltages will displace the mass until it reaches a stable equilibrium due to the spring's resistance, as seen in Figure 3(b). Voltages applied above a specific threshold generate a force strong enough to pull the mass into contact with ground, as seen in Figure 3(d). This threshold voltage is known as the pull-in voltage, V_p. The pull-in voltage can be solved for by finding the static system's total stiffness. The point of zero stiffness, where an unstable equilibrium exists, occurs where the mass is at one third of the distance across the gap as in Figure 3(c). Solving for the voltage required for a displacement of *d/3* gives the following:

$$V_{app} = V_p = \sqrt{\frac{8}{27}\frac{kd^3}{\varepsilon_0 A}} \tag{2}$$

For more detail on the derivation of the unstable equilibrium position and the pull-in voltage for static displacement, see the referenced literature[22].

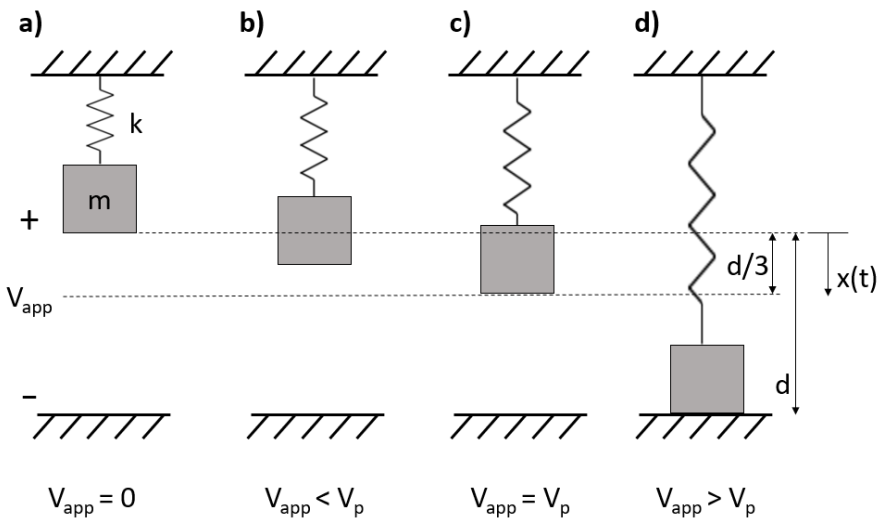

Figure 3: Equilibrium positions in the simplified Spring-Mass System

1.4 The Doubly-Clamped Beam Design

1.4.1 Description of Results and Design process

Previous work was done to design an ESD device that meets the above specified design requirements, but the devices failed to demonstrate repeatable behavior[19]. The design consisted of a rectangular single-layer graphene nanoribbon suspended across an etched trench, seen in Figure 4.

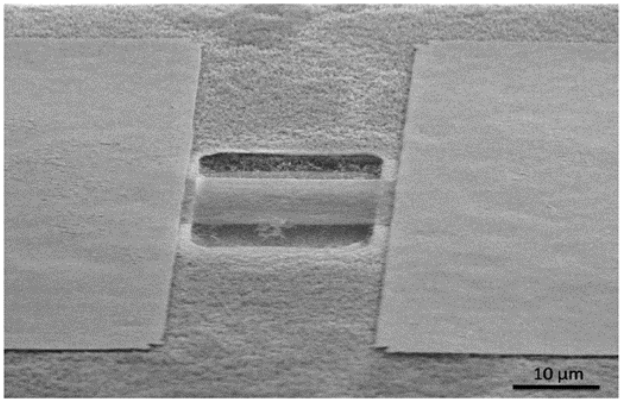

Figure 4: SEM image of a previously designed and fabricated graphene ESD nanoribbon (image courtesy of Jimmy Ng)

The graphene ribbon in this design was modeled as a rectangular cross-section beam under a uniformly distributed load with a clamped condition at each end. The pull-in voltage is solved for using Equation 2 of the previously described SDOF model, where A is the area of the bottom of the ribbon. A lumped spring constant for the entire ribbon, k, can be derived from the corresponding Bernoulli-Euler beam equation for maximum deflection:

$$\delta_{max} = \frac{wl^4}{384EI} \tag{3}$$

where δ_{max} is the maximum deflection, w is the distributed load, l is the length of the beam, E is the Young's modulus of the beam's material and I is the beam's moment of inertia[23]. Based on Hooke's Law, the applied load, wl, was divided by the equation for δ_{max} to find an effective spring constant, k, for the design. Based on experimental data from testing of the doubly clamped nanoribbon design, a factor of 4900 was multiplied into the spring constant:

$$k = 4900\left(\frac{32Ebh^3}{l^3}\right) \tag{4}$$

The experimentally-fitted factor accounts for several effects not accounted for in the SDOF beam model. These effects include beam stiffening due to a thick residual PMMA layer that cannot be completely removed from the graphene during fabrication, doping of the graphene caused by the aforementioned residual polymer layer and stiffening due to ripples within the graphene ribbon[13]. Other effects are limitations of the model in not accounting for nonlinear stiffness in the beam or variation in the capacitance along the ribbon as displacement occurs.

Devices were designed for specific V_P values by varying k, A and d in Equation 2. Testing yielded the desired behavior on the first pull-in cycle, but resistor-like behavior on subsequent cycles, indicating that the ribbon must have slipped from between the dielectric and metal layers and shorted to ground[24]. A modified design was created to secure the graphene ribbon. Two metal "nails" were created by etching holes into the fixed ends of the ribbon and then depositing a metal layer above the graphene layer. Experimental results for this modified device yielded the expected pull-in voltage on the first cycle and then decreasing pull-in voltages on subsequent cycles until again exhibiting resistor-like behavior[19]. The decreasing pull-in voltage results from high stress at the "nails" and in the PMMA layer. Yielding or fracture of the PMMA or the graphene ribbon would lower the ribbon's stiffness until flexible enough to be held down by van der Waals forces.

1.4.2 Analysis of Repeatability Issues Using FEA

Evidence of these issues is seen by comparing the tested behavior of a 5μm by 7μm ribbon with the simulation results for a finite element graphene-PMMA composite ribbon of the same dimensions. The simulation results, shown in Figure 5, predicted stresses of up to 65MPa in the PMMA and 5.13GPa in the graphene, exceeding the PMMA tensile strength of 62MPa and

approaching the SCG fatigue strength for 40GPa. The PCG fatigue strength has not been determined but is likely lower since its fracture strength is about one-third of SCG's.

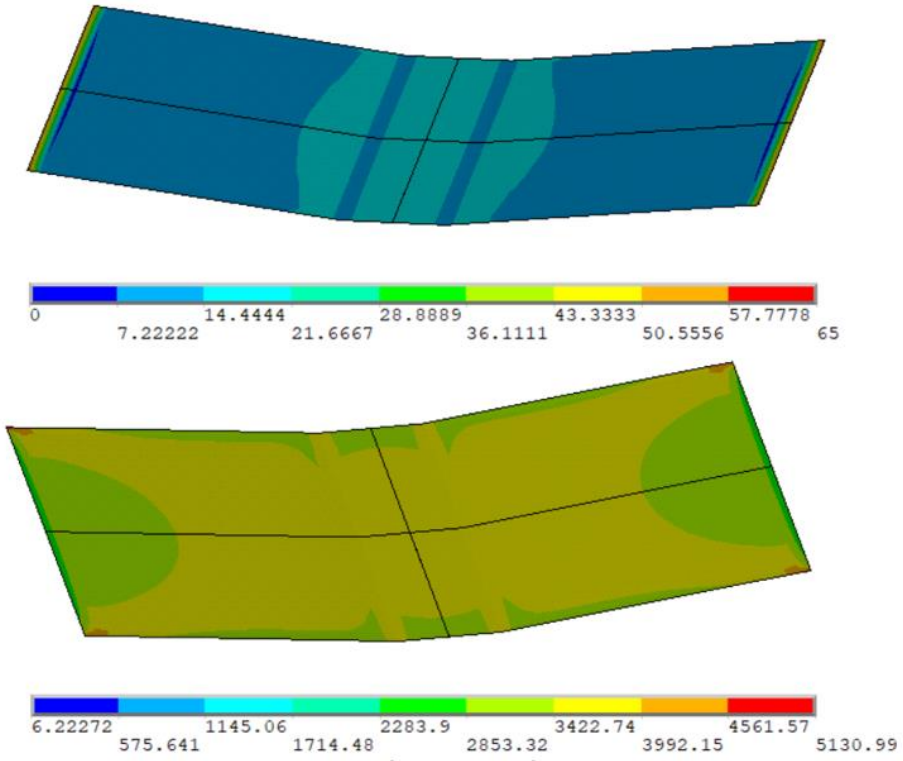

Figure 5: Von Mises stresses induced at contact in the PMMA layer (top) and the graphene layer (bottom) of a 5μm by 7μm beam. All units are in MPa.

Additionally, the reaction forces at the fixed end condition measured 3.10μN in the direction away from the fixed condition. Pull-in with the PMMA layer occurred at 9.75V but when the PMMA layer was removed from the simulation, it occurred at only 5.97V. The fabricated ribbon exhibited an 8V pull-in on the first cycle, but subsequent cycles resulted in progressively lower pull-in voltages until degrading into resistor-like behavior by the eighth cycle.

Chapter 2

Design of New ESD Devices Using a Simplified Beam Theory Approach

This chapter describes the process used to determine the 2D geometry and dimensions of new designs for a graphene nanoribbon ESD device. The methods for determining the shape of the new designs is first described. Next is a description of the theory used to optimize the dimensions within the fabrication limitations. Limitations in the theory are acknowledged, but results are found insightful in improving upon previous designs. The designs are also shown to satisfy the electrical design requirements.

2.1 Lamina-Emergent Graphene Nanoribbon Mechanism Designs

The focus for the new designs is to produce ribbon designs with repeatable behavior by reducing stresses in the ribbon and reducing reaction forces at the fixed boundaries while maintaining a high pull-in voltage. High stresses can lead to material failure in the PMMA, reducing the device's spring constant over time. High reaction forces can cause slipping or high stress concentrations in the presence of "nails". These two goals can be achieved by introducing flexures that combine to function as a lamina emergent mechanism that is free to displace under electrostatic loads[25].

The FACT theory of compliant mechanism design was used to help identify possible geometries for SDOF translation in the direction of pull-in[26]. Within the design space of a 2D ribbon of material, however, a solution containing flexures that exist within two parallel planes

was not found. Since actuation of the device is governed by a low frequency downward electrostatic force, a set of flexures in a single plane was deemed appropriate. The three chosen designs, seen in Figure 6, are named 1Blade (1B), 2Blade (2B), and Torsion-Supported (TS). Each consists of flexures meant to behave as springs as the device is actuated, and a large suspended stage meant to maximize the discharge area that contacts the trench bottom. The strain experienced by the ribbon during actuation is mostly located within the flexures, reducing the tugging effects at the boundary conditions and distributing the stress across more ribbon area.

The 2B design draws on ideas from existing orthoplanar spring designs[27]. The four double-length flexure arms allow for large displacements. Equal lengths in the two arm segments minimize in-plane rotation of the stage that results in screw motion during displacement. Having four flexure arms maximizes the spring constant in the rectangular device footprint while reducing undesired stage rotation by having support points at each corner of the stage.

The 1B design is a modification of the 2B design that attempts to reduce underconstraint in the ribbon motion. Single length flexure arms eliminate the extra mass in the middle of the flexure that could result in poor dynamic behavior.

The TS design is an attempt to maintain the high effective spring constant of the fixed-fixed beam design by rotating the beam to bend along the trench and replace translational reaction forces with torsional reactions. Four small sections of graphene ribbon are added to function as torsion springs for the ribbon to hinge on. More flexible torsion sections would approach the behavior of a simply supported beam while less flexible torsion sections would approach the behavior of a fixed-fixed beam.

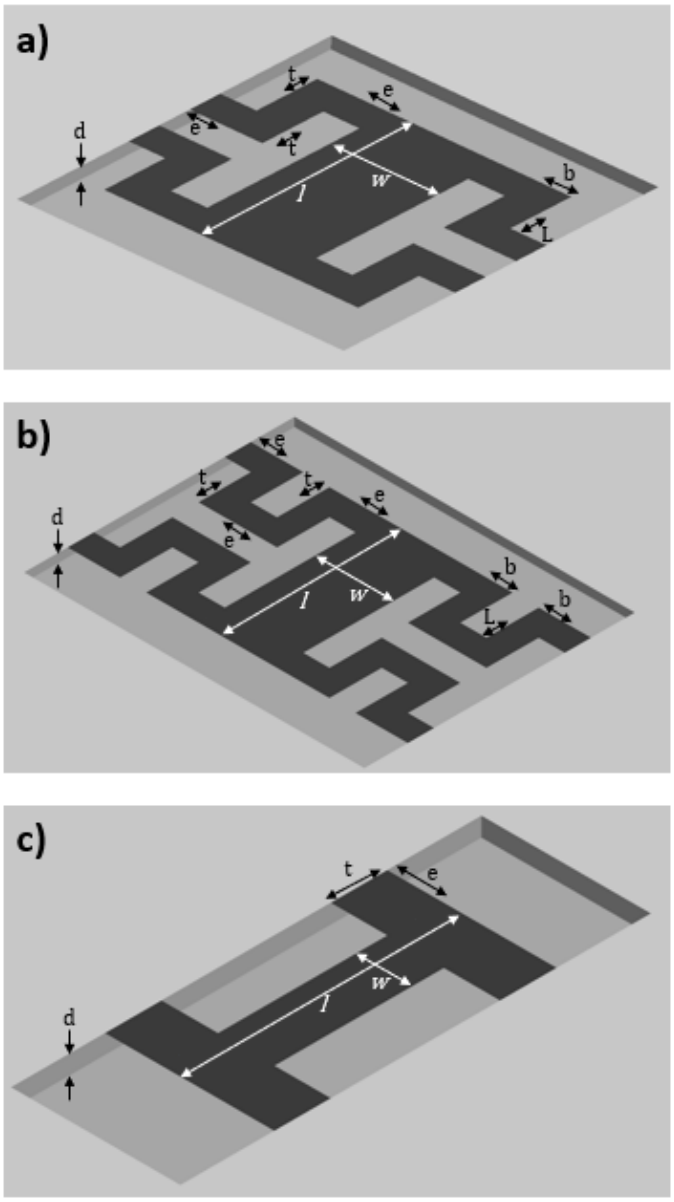

Figure 6: The (a) 1B, (b) 2B and (c) TS ribbon designs

2.2 Static Analysis and Optimization Using the SDOF Beam Model

For an initial analysis of each ribbon design, the spring-mass model described in Chapter 1 was used with Bernoulli-Euler beams and the experimentally determined fitting factor from the previous design to guide the design process. Although limitations exist in the way this approach represents the behavior of a graphene nanoribbon, the approach allows the designer to optimize

the dimensions of each ribbon's features for a high pull-in voltage. Using Equation 2, the authors identified the values of *k*, *d*, and *A* giving the highest pull-in voltage for each design within the fabrication limits. FEA methods used later for a more accurate prediction of pull-in voltage will be described in Chapter 3.

The value of *A* used for each design is the area of the discharge stage. For each design, this is the product of *w* and *l* as seen in Figure 6.

The *k* values for the 1B and 2B designs were determined by deriving an effective spring constant for each flexure arm using Castigliano's method. The discharge stage is treated as a rigid body and the flexures are treated as curved beams supporting the discharge stage. In using Castigliano's method, a generic placeholder force, represented as *F* in Figure 7, can be applied to the stage end of the flexure.

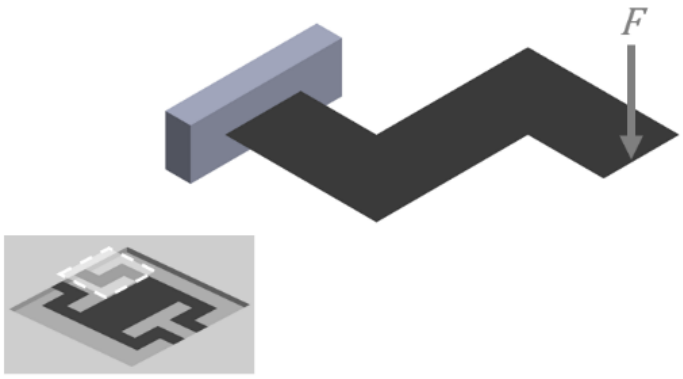

Figure 7: Castigliano's method for curved beams applied to the flexures of the 1B design

The partial derivative of the strain energy in each segment of the flexure is then calculated with respect to the placeholder force, giving the displacement in terms of the force:

$$y_j = \frac{\partial u}{\partial F} = \sum_{j=1}^{m} \int \frac{M_j}{E_j I_j} \frac{\partial M_j}{\partial F} dx + \sum_{j=1}^{m} \int \frac{T_j}{G_j J_j} \frac{\partial T_j}{\partial F} dx \quad (5)$$

Each integrated term in the above equation represents the displacement due to the strain energy, u, induced in each of the m number of segments of the flexure by bending moment loads, M_j, and torsion loads, T_j, caused by the placeholder force. Shear force loads are ignored due to their negligible contribution to the result. More details on using Castigliano's method can be seen in the reference[23]. Dividing the total displacement by the placeholder force gives the spring constant for each flexure. The four flexures are then treated as springs in parallel and the total spring constant for the entire device is calculated:

$$k = C_{exp} k_{tot} = C_{exp}(k_1 + k_2 + k_3 + k_4) \tag{6}$$

where k_1 through k_4 represent the spring constants of the four flexures and are all equal. The effective spring constant, k, of the device is then calculated by multiplying k_{tot} by the experimentally determined factor, C_{exp}. The TS design is assumed to behave similarly to a simply-supported beam, and its effective spring constant was derived from the simply-supported beam deflection formula. The complete expressions for k of each design can be seen in the MATLAB scripts in Appendix A.

Most dimensions within the design space easily satisfied the design requirements other than pull-in voltage. For the 1B and 2B designs, a pull-in voltage of 1.5V was unachievable without passing the minimum feature size. MATLAB's constrained nonlinear multivariable function minimizer was therefore used to optimize each design's *k*, *d*, and *A* parameters for maximum pull-in voltage value by minimizing the value of V_p^{-2}. Limits on *w* and *l* were set to give flexures space to avoid collision during actuation. The optimal pull-in voltages come from the cases where discharge stage area, A, is minimized while spring constant, k, and trench depth, d, are maximized.

The minimum feature size of 3μm limits the minimum achievable platform area and the achievable spring constants in each design. The optimal trench depth is similarly at its fabrication limit. The optimized dimensions for each design are listed in Table 1.

Table 1: Optimized dimensions and predicted performance for designs shown in Figure 6

Design	**1B**	**2B**	**TS**
d (μm)	1	1	1
l (μm)	21	21	15
w (μm)	9	9	3
L (μm)	3	3	-
b (μm)	3	3	-
t (μm)	3	3	3
e (μm)	3	3	3
k (N/m)	2.48×10^{-4}	6.40×10^{-5}	6.57×10^{-4}
V_P *(V)*	0.209	0.107	0.699

2.3 Dynamic Analysis Using the SDOF Beam Model

One assumption made in the calculation of pull-in voltage using Equation 2 is that inertial forces do not come into play. When the stage starts at its unstable equilibrium position of one-third of the trench depth with no initial velocity, this assumption is valid. In such a case, the discharge stage will remain stationary when V_p is applied. However, a more realistic situation for the application in question involves a platform that is either already moving or starts from rest at the zero position when the pull-in voltage is applied. In this case, the inertial forces would act against the spring forces that hold the mass in its equilibrium position, sending the stage past the unstable equilibrium and into the region where the electrostatic forces dominate. A "dynamic pull-in voltage", that defines the voltage that causes pull-in from zero initial conditions, is therefore needed to more accurately characterize each design's behavior.

To calculate the dynamic pull-in voltage, the parameters for each design are substituted into the differential equation that governs the system, Equation 1. The mass is calculated by taking the area of the ribbon's stage and multiplying by the 2D density of single-layer graphene, 0.77×10^{-6}kg/m^2 [28]. A MATLAB script is then used to simulate the motion of the system until either pull-in occurs or an equilibrium is reached. The script runs at different voltages until it converges on the lowest voltage at which pull-in will occur. This is the dynamic pull-in voltage. The plots in Figure 8 illustrate the response of a discharge stage when a stable equilibrium is reached and when pull-in occurs.

The same technique is also applied to predict the mechanical response time of each design. Figure 8 shows, from left to right, the response times of the 1B design when a high voltage is applied, when the pull-in voltage is applied, when the dynamic pull-in voltage is applied, and when a voltage just below the dynamic pull-in voltage is applied.

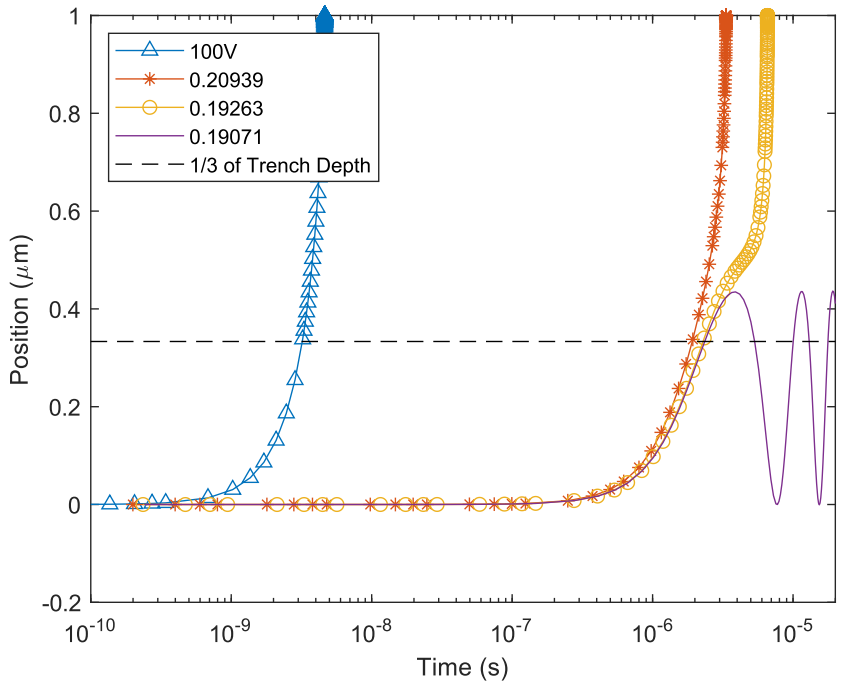

Figure 8: Response of the 1B ribbon design when various voltages are applied. The vertical axis plots the discharge stage position in the positive x direction seen in Figure 3.

The motion of the 1B stage in Figure 8 closely reflects the responses for the 2B and TS designs. Response times on the order of nanoseconds are only achievable at high voltages. The response times at the pull-in voltage are on the order of microseconds. Faster response times could be achieved with lower spring constants or a larger stage, but both changes would adversely affect the pull-in voltage. Under different fabrication limitations, these two parameters along with trench depth would have more freedom and could improve the mechanical response time.

Also observable in the plot is the frequency response for the 1B design. Using the spring-mass model, the natural frequency can be calculated from the spring constant and the mass:

$$f_n = \frac{1}{2\pi}\sqrt{\frac{k}{m}} \tag{7}$$

where m is the mass of the discharge stage and k is the effective spring constant[29]. Table 2 lists the natural frequency for each design.

Table 2: Predicted dynamic behaviors

Design	**1B**	**2B**	**TS**
Dynamic Pull-in Voltage (V)	0.191	0.0980	0.643
Response Time at Dynamic Pull-in Voltage (s)	6.58×10^{-6}	1.29×10^{-5}	1.97×10^{-6}
f_n *(Hz)*	2.08×10^{5}	1.06×10^{5}	6.93×10^{5}

The natural frequency is only of concern after the stage is released from ESD contact or after a voltage lower than the dynamic pull-in voltage is applied. After each of these events, the stage oscillates around the stable equilibrium position until a new voltage is applied, at which point the equilibrium position will shift according to the voltage. If any voltage is cyclically applied at the natural frequency, resonance could occur, causing the amplitude of the oscillation to increase until material failure or until the stage enters the region where electrostatic force dominates. A situation where resonance causes the stage to enter the electrostatic-force dominated region would therefore cause pull-in at voltages below the dynamic pull-in voltage. The applications listed in the introduction section are not expected to induce charges more than every few seconds. Any natural frequency close to the MHz range would therefore be more than satisfactory.

2.4 Electrical Design Requirements

As mentioned in previous sections, each design was optimized for the highest achievable pull-in voltage due to fabrication constraints. The two electrical design requirements were met without being optimized. Since all parameters are fixed other than those determined by the design's geometry, the electrical time response requirement can be simplified to:

$$\frac{A^2}{d} < 4.34 \times 10^{-10}\text{m}^3 \quad (8)$$

where A is the area of the discharge stage and d is the trench depth. The parasitic capacitance requirement can be simplified to:

$$\frac{A}{d} < 1.69\text{m} \quad (9)$$

where the A and d are defined as above. Calculations for the electrical time response and parasitic capacitance requirements for each design can be seen in Table 3.

Table 3: Calculated values for electrical design requirements

Design	**1B**	**2B**	**TS**
Electrical Time Response Requirement (m^3)	3.572×10^{-14}	3.5721×10^{-14}	2.03×10^{-15}
Parasitic Capacitance Requirement (m)	1.89×10^{-4}	1.89×10^{-4}	4.50×10^{-5}

2.5 Dynamic Analysis Using the FACT MDOF Beam Model

After optimization based on the SDOF model, a multiple-degree-of-freedom (MDOF) analysis was performed based on the FACT flexure design theory. This model was expected to provide more insight regarding the motion of the system. The theory is still based on the stiffness of Bernoulli-Euler beams and therefore, there are still limitations in how it models the likely nonlinear behavior a graphene ribbon. The experimentally determined factor, C_{exp}, is still used in this approach to compensate for the additional stiffness seen experimentally.

FACT requires modeling the ribbon as massless flexures connected in series and parallel by rigid stages. All three designs can be classified as hybrid mechanisms. Each ribbon design was broken into distinct flexures and rigid bodies, as shown in Figure 9. To more accurately capture flexibility at the discharge stage, it is broken into two large flexures, F1 and F2, with a rigid body, S1, of length $l/1000$ in the center. When calculating the electrostatic forces that act on the discharge stage and the intermediate stages between flexures, an effective area is assigned to each stage. For all stages, half of the area of the adjoining flexures is added to the stage's area to calculate the effective electrostatic area. As an example, for stage 1, the discharge stage, half the area of flexures 1 and 2 are added to the area of stage 1.

Figure 9: Definition of stages (black) and flexures (gray) in each design for use with FACT.

Whereas the SDOF system could be modeled according to Equation 1, the dynamic MDOF system must be modeled using screw theory kinematics as:

$$\boldsymbol{W} = [M_{TW}]\ddot{\boldsymbol{T}} + [K_{TW}]\boldsymbol{T} \tag{10}$$

where, if b is the number of rigid stages, then $\boldsymbol{W}$ is a $6\mathrm{b} \times 1$ wrench vector containing the forces applied to each of the stages, M_{TW} is a $6b \times 6b$ mass matrix, K_{TW} is a $6b \times 6b$ stiffness matrix that is multiplied by a factor of C_{exp} and $\boldsymbol{T}$ is the $6b \times 1$ twist vector representing the degrees of

freedom of each stage. Details on the setup of such a system can be found in the references on FACT[26,30,31]. For each stage, i, in the system, the applied force in the z direction, W_{6i-3}, is:

$$W_{6i-3} = -\frac{\varepsilon_0 A_i V_{app}^2}{2(d - T_{6i})^2} \tag{11}$$

where A_i is the area of that stage and T_{6i} is the z direction degree of freedom for that stage in $\boldsymbol{T}$. Here, the z direction refers to the $-$x direction seen in Figure 3. All other elements of $\boldsymbol{W}$ are equal to 0.

Appendix D shows how MATLAB was used to set up the dynamic system. The resulting pull-in voltages, response times and natural frequencies can be seen in Table 4. Differences can be seen between the SDOF and MDOF models, but the 2B design maintains the lowest pull-in voltage while the TS design exhibits the highest pull-in voltage. The slightly higher pull-in voltages reflect two changes in the modeling technique: (1) the largest area exposed to electrostatic forces is treated as multiple bodies instead of one body that exists at the point of maximum displacement in the whole ribbon, and (2) the length of the some of the flexures is incorporated into rigid bodies. The significantly higher V_p for the TS design shows that the simply supported beam model used in the SDOF analysis allows for too much flexibility.

Table 4: Predicted MDOF dynamic behaviors, including the three lowest natural frequencies

Design	**1B**	**2B**	**TS**
Pull-in Voltage (V)	0.265	0.117	1.24
Response Time at Pull-in Voltage (s)	1.46×10^{-9}	3.66×10^{-9}	4.67×10^{-10}
f_n *(Hz)*	3.34×10^{8} 3.82×10^{8} 6.33×10^{8}	2.21×10^{8} 2.67×10^{8} 3.27×10^{8}	2.74×10^{9} 6.63×10^{9} 7.29×10^{9}

The MDOF response times are also much quicker, and the natural frequencies are much higher. Additionally, the MDOF analysis shows that all portions of the ribbon collapse to the trench bottom, rather than just the stage collapsing while the flexures remain suspended. Figure 10 shows the response for a 2B ribbon, where stages 8 and 15 are predicted to arrive at the trench bottom just before the rest of the stages, including the discharge stage.

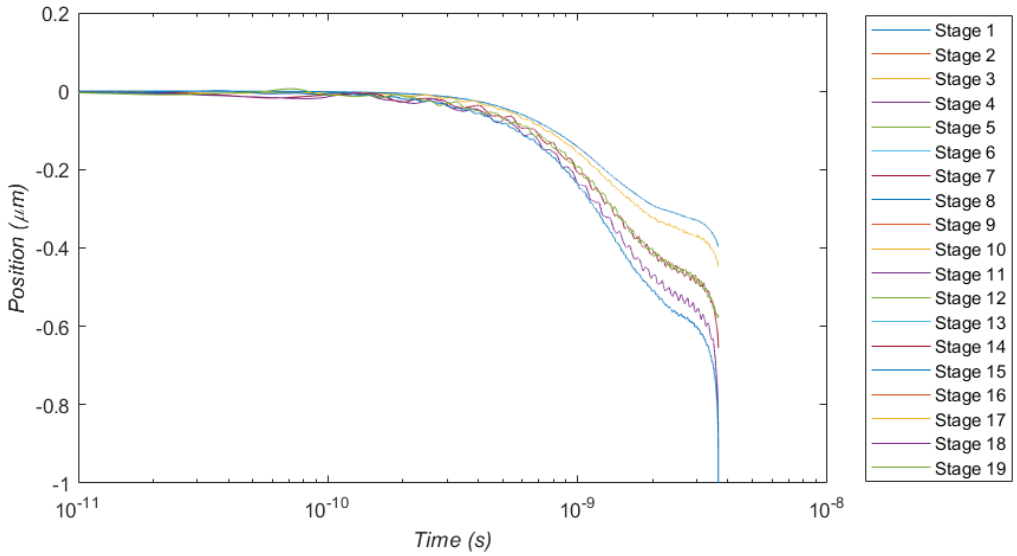

Figure 10: Response of a 2B ribbon at its pull-in voltage modeled using FACT

The natural frequencies agree with 2B being the most flexible and TS being the stiffest, as they have the lowest and highest natural of the three designs. The corresponding mode shapes for these nine frequencies represent a combination of the stages moving in the z direction. For the TS design, the first frequency corresponds to a mode shape of all three stages moving down together with the flexure stages closely trailing the discharge stage. For the 2B design, the second and third frequencies have all stages moving down together with the flexures slightly leading the discharge stage. The first frequency shows the flexure stages moving slightly up and down opposite each other. The 1B design exhibits similar mode shapes but with the discharge stage leading the flexure

stages in the second frequency. These results suggest two phenomena not captured in the SDOF model: (1) large rippling effect in the flexures due to underconstraint that is also seen later in the nonlinear FEA and MD models, and (2) low stiffness throughout the ribbon causing the whole ribbon to contact the trench bottom under small trench depth conditions.

Chapter 3

Analysis of Devices Using Finite Element Software

For more accurate predictions of pull-in voltage, as well as an analysis of stresses and reaction forces, the ANSYS Mechanical APDL software package is used. Input scripts used can be found in Appendix B. Each design is modeled using SHELL281 elements and settings for nonlinear geometry are enabled. Since the fabrication process produces a PMMA layer of an average thickness of 4.12nm which cannot be completely removed, this layer is included above the 0.34nm thick graphene layer. The experimentally determined fit factor is not included in this analysis. To save computation time, each design is cut in half along the stage's length and then again cut in half along its width. Symmetric boundary conditions are applied to the two cut edges of the stage. The fixed ends of each flexure are fixed in all rotational and translational degrees of freedom.

3.1 Pull-in Voltage Validation

Electrostatic forces are simulated by setting the applied voltage and ground voltage to the top and bottom, respectively, of an "elastic air" region. The elastic air region is created from SOLID226 bricks that fill the volume between the trench floor and the graphene ribbon, seen below the graphene-PMMA shell elements in Figure 11. ANSYS uses a parallel plate capacitor model, as described in section 2.9 of the release 18.0 manual, to determine capacitance. To solve for the pull-in voltage, the applied voltage is progressively increased during load steps of a static analysis until the solver fails to converge on a solution. Failure to find a solution indicates that the

electrostatic forces have overcome the ribbon's spring forces and pull-in has occurred. A plot of the equilibrium position for each ribbon design at various voltages can be seen in Figure 12.

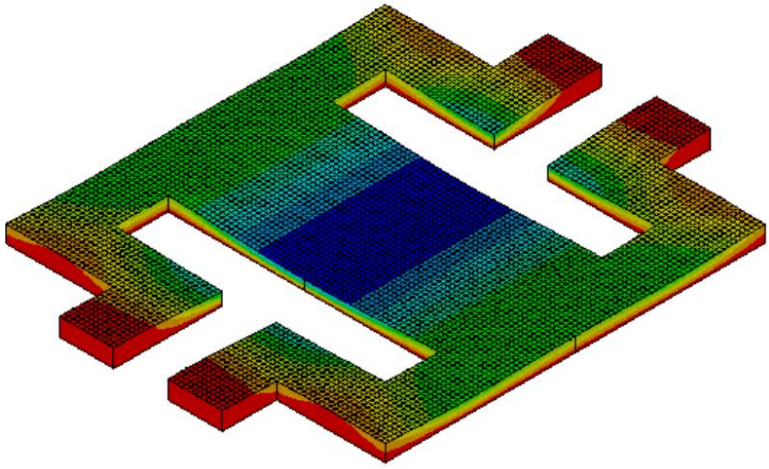

Figure 11: Displacement plot of a 1B ribbon under an electrostatic load. Results have been reflected about its symmetry loads for a visual of the complete device. Displacement ranges from 0 (red) to 0.454μm (blue).

Table 5 shows the pull-in voltage for each design with and without a PMMA residue layer. It is interesting to note that the polymer residue has a greater effect on the TS design and the fixed-fixed design than on the other two. This effect indicates that the design will be more adversely affected by polymer fatigue.

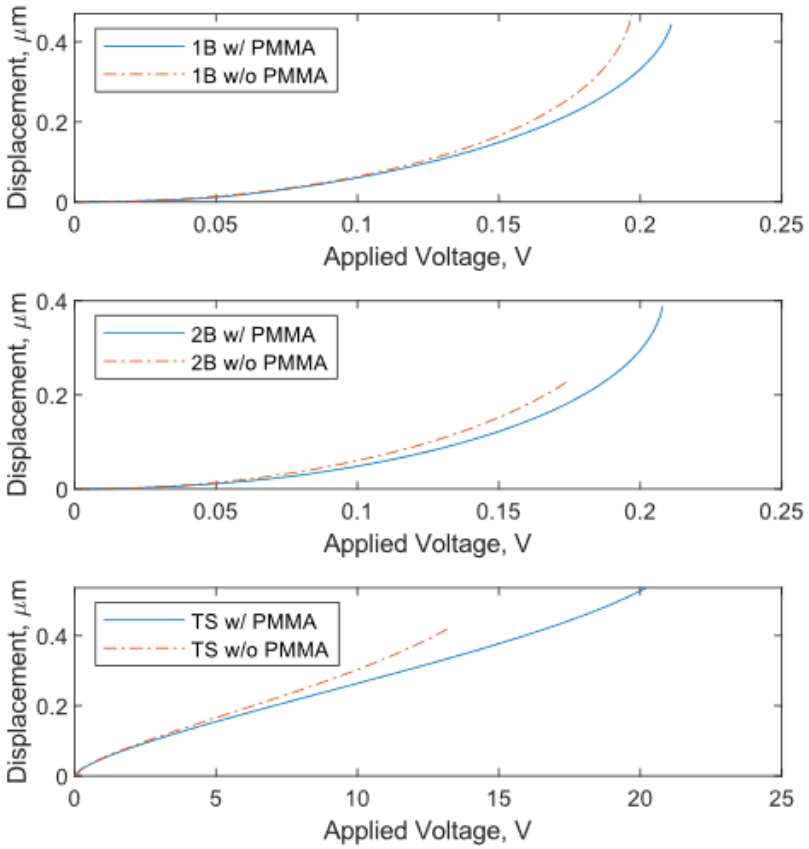

Figure 12: Equilibrium position up until the pull-in voltage of each ribbon design

3.2 Repeatability Analysis Using Finite Element Modeling

Since the above method cannot solve for the ribbon's state once it has passed its unstable equilibrium position, a separate model was created. In this model a displacement load of d was applied across the middle of the stage in each ribbon design. As seen in Figure 13, filleted corners were added into the model for the stress analysis. These fillets of a 1μm radius are an effect of the etching process, as seen in Figure 28, but also serve as stress relievers. Plots of the von Mises stress were obtained for each design and the reaction forces acting toward the toward the stage at the fixed boundary were totaled. Results are shown in Table 5.

Table 5: Pull-in Voltages, Maximum Von Mises Stresses and Reaction Forces

Design	**1B**	**2B**	**TS**
Pull-in Voltage with PMMA (V)	0.212	0.208	20.2
Pull-in Voltage without PMMA (V)	0.198	0.175	13.3
Maximum Stress on PMMA (MPa)	1.25	0.85	33
Maximum Stress on Graphene (MPa)	54.07	43.3	9630
Reaction Force (μN)	1.24×10^{-4}	1.13×10^{-4}	1.52

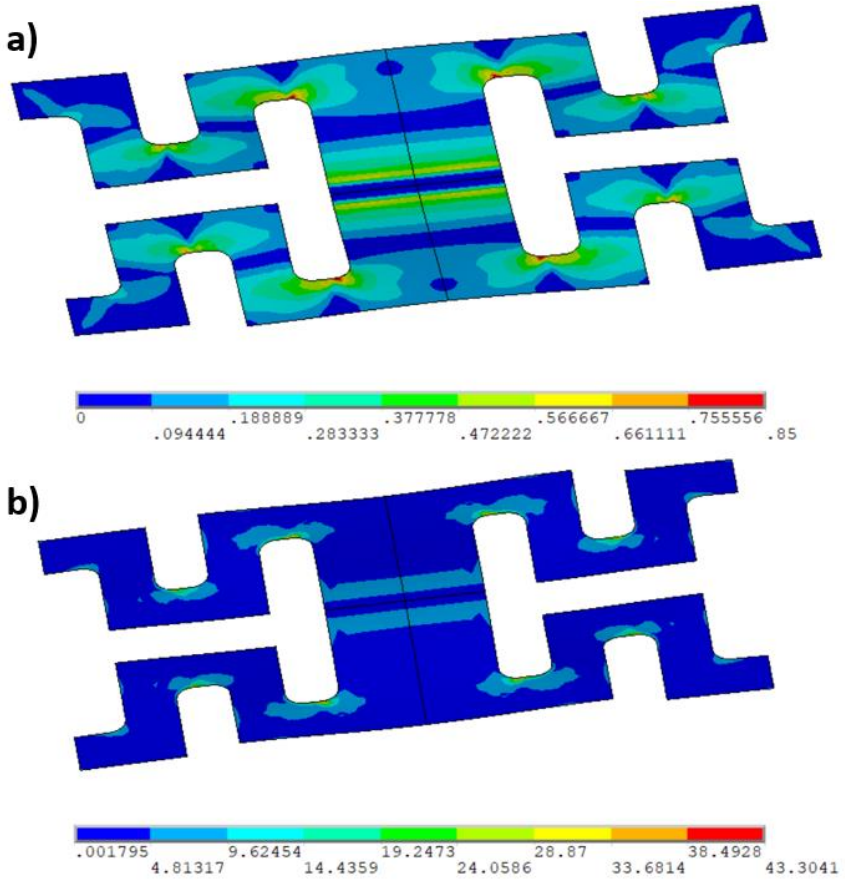

Figure 13: Von Mises stress plots of the PMMA side (a) and the graphene side (b) of the 2B ribbon under a 1μm displacement load. All units are in MPa.

Chapter 4

Atomistic Studies

This portion of the work used computational and storage services associated with the Hoffman2 Shared Cluster provided by UCLA Institute for Digital Research and Education's Research Technology Group.

To validate the finite element simulations of graphene's mechanical behavior, atomistic simulations were performed using Sandia National Laboratory's LAMMPS (Large Atomic/Molecular Massively Parallel Simulator) software[5]. This chapter describes methods used to determine the pull-in voltage of a scaled down version of each design. The input scripts can be found in Appendix C. Then, results are compared to a finite element analysis on the same scale.

A variety of studies have used LAMMPS to simulate graphene and have found its results in agreement with experimental results[3,12,40–43,32–39]. Due to the computational cost of molecular dynamics (MD) simulations, each design had to be scaled down to nanometer dimensions as listed in Table 6. A minimum feature size of 30Å was chosen based on the minimum feature size in similar studies[12,40]. Simulations at sizes within the fabrication limits would provide more accurate insight but could not be achieved without access to better resources. The residual PMMA layer that exists on the devices also could not be included in the model for the same reasons.

Table 6: Dimensions used for LAMMPS simulations

Design	**1B**	**2B**	**TS**
l (Å)	210	210	150
w (Å)	90	90	30
L (Å)	30	30	-
b (Å)	30	30	-
t (Å)	30	30	30
e (Å)	30	30	30

For each simulation, carbon atoms are placed 1.42Å apart on a hexagonal lattice to form the geometry corresponding to the 1B, 2B, or TS design, shown in Figure 27[10]. The AIREBO (Adaptive Intermolecular Reactive Empirical Bond-Order) potential is used with a carbon-carbon cutoff distance, R_{CCmin}, of 1.92Å to create a pairwise force field between each atom. The default cutoff distance of 1.7Å causes unnatural behavior between carbon atoms when the graphene reaches strains close to fracture[37,44]. Visual renderings of simulations are generated using the OVITO (Open Visualization Tool) software[6].

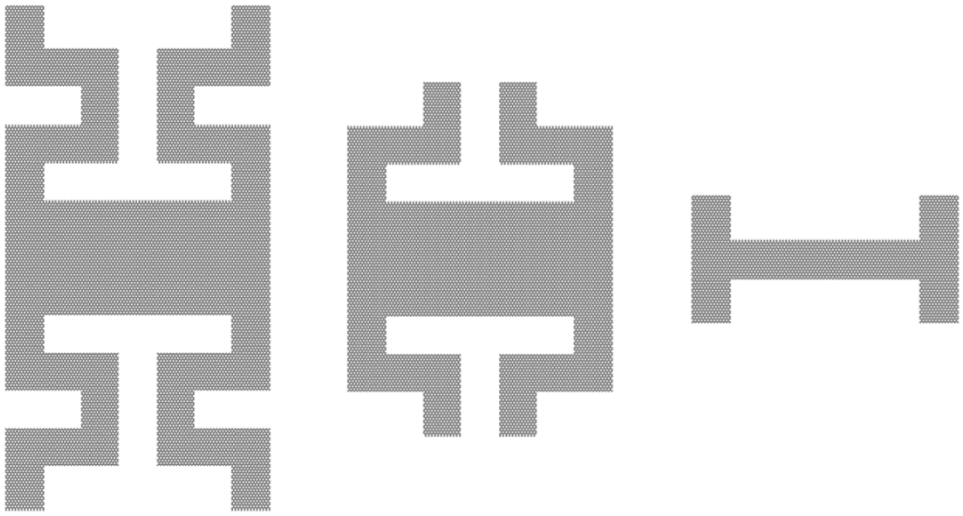

Figure 14: Carbon atoms placed on a hexagonal lattice to form graphene nanoribbons and cut according to the 1B (left), 2B (center) and TS (right) designs

4.1 Dynamic Equilibrium of Designs

To properly study the dynamic behavior of graphene, it is necessary to allow the atoms to reach a state of dynamic equilibrium. Graphene naturally forms out-of-plane ripples due to the motion caused by thermal energy in the individual atoms. In the equilibrium setup, fixed simulation boundaries are applied in the x and y directions to encompass the 2D ribbon geometry. A shrink-wrapped boundary is applied in the z direction that grows or shrinks according to the magnitude of the out of plane ripples. The NVT (constant Number of atoms, system Volume and system Temperature) ensemble is used where temperature is held constant at 300K throughout the simulation. Simulations are run at a timestep of 1fs[12,40].

To enforce the fixed-end boundary conditions on the graphene ribbons, two groups of atoms were created: the fixed graphene group and the mobile graphene group. The mobile graphene group consists of all the atoms within the dimensions shown in Table 6. The only forces acting on the mobile atoms are those that result from the AIREBO energy potential between carbon atoms and the thermal motion. The fixed group has a zero force imposed on each atom which prevents motion in any direction.

The total potential energy in the system, the total kinetic energy of the system, the number of atoms, and the positions of certain atoms as well as average positions of groups of atoms were tracked each timestep and read into MATLAB for post-processing. To determine when the graphene ribbon had reached a dynamic equilibrium state, the potential energy was divided by the total number of atoms and then plotted against time, as shown in Figure 15. These per atom energy averages were used to conveniently compare values between designs having different numbers of atoms. Dynamic equilibrium was reached when the potential energy per atoms oscillated around a

constant value, listed in Table 7. For the 2B design, dynamic equilibrium was reached after approximately 200ps.

Table 7: Results of the dynamic equilibrium simulation. Extreme atom positions are given in distance from the plane where the fixed ends are held.

Design	**1B**	**2B**	**TS**
Extreme Atom Position (Å)	40	50	5
Extreme Average Atom Position (Å)	3	4	0.7
Extreme Stage Atom Position (Å)	29	21	5
Extreme Average Stage Atom Position (Å)	10	7	0.7
Equilibrium Potential Energy (eV/Atom)	-7.2536	-7.2387	-7.2402

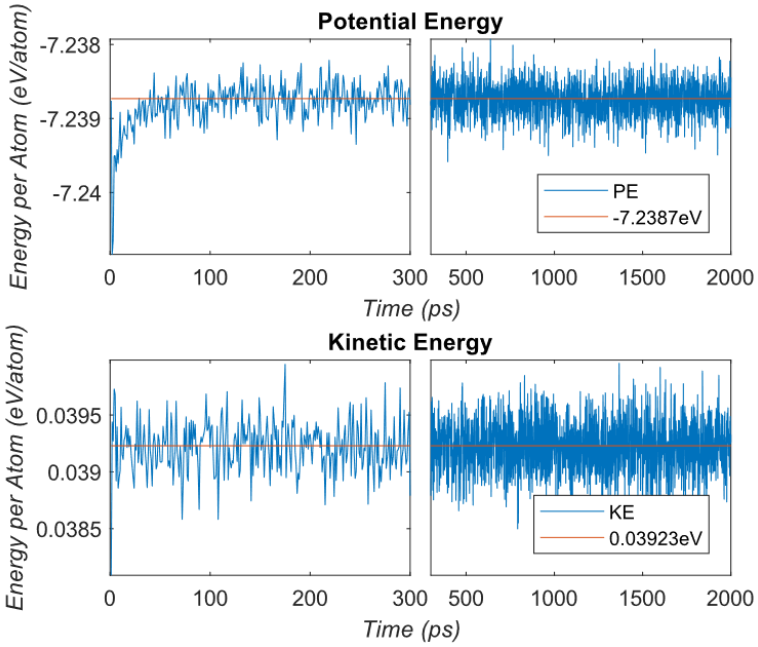

Figure 15: Per atom energy as dynamic equilibrium is reached for a 2B graphene nanoribbon

To further study the behavior of each design, the minimum, maximum and average atom positions were plotted over time. Atom positions in a discharge stage area were also plotted to

understand the mode shapes of the thermal oscillation. A discharge stage was defined for each design as follows: the 1B stage contains the whole rectangular area supported by the four flexures, the 2B stage contains the whole rectangular area supported by the four flexures and the TS stage contains the center half of the rectangular area supported by the four flexures. Figure 16 shows that a regular oscillatory behavior begins around the same timestep that the potential energy stabilizes. Table 7 shows the most extreme value across all timesteps for the each of the positions being tracked.

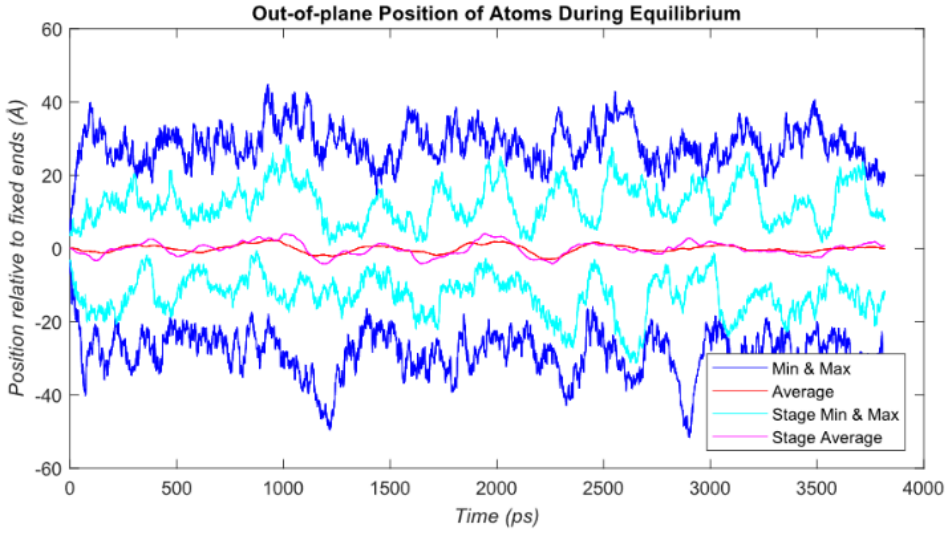

Figure 16: Minimum, maximum and average atom displacement in a 2B ribbon over time

4.2 Pull-in Simulation

To simulate electrostatic pull-in, the atom positions, velocities and forces were read in from a simulation snapshot created during the dynamic equilibrium simulation. Figure 17 shows the state of each design at the timestep used for this snapshot.

Figure 17: Renderings of the 1B design after 2400ps (left), 2B design after 1800ps (center) and TS design after 1000ps (right). Each atom is colored according to its z position.

A simulation box fixed in all three dimensions is created to encompass the graphene ribbon and a reflective wall is created at the bottom surface of the simulation box, representing the trench bottom. Van der Waals attraction to the trench bottom can be simulated by applying a 12-6 Lennard-Jones potential to the wall where a silicon-carbon interaction is described by an ϵ value of 0.00891eV and a σ value of 2.63Å[45]. This implementation was removed, however, due to its small effect and to better reflect the conditions in the FEA simulations for comparison.

Since thermal oscillations cause large amounts of displacement without the presence of external forces, the trench is made sufficiently deep to avoid pull-in caused by displacement experienced during equilibrium. There are two modes by which thermal oscillation could cause pull-in: (1) when the trench depth is smaller than the maximum displacement experienced by a carbon atom and (2) when momentum in the atoms causes the ribbon to move past an unstable equilibrium position. The first case occurs without any electrostatic force applied and the second could occur with a weak electrostatic force applied. Based on the $d/3$ unstable equilibrium position described in Chapter 1, a trench depth of at least three times the most extreme atom position experienced during equilibrium was deemed sufficient to differentiate between thermally-induced pull-in and electrically-induced pull-in. The simulations are again run at a 1fs timestep and are performed under the NVT ensemble where the temperature is held constant at 300K.

An electrostatic force is applied to each atom until the lowest atom is within 1Å of the trench bottom, indicating pull-in. The applied force is derived from the behavior of a parallel plate capacitor under a constant voltage, as is done in the ANSYS simulations. Each atom is treated as a movable plate containing a charge, Q, of:

$$Q = CV = \frac{\varepsilon_0 A_{atom} V_{app}}{d} \tag{12}$$

where ε_0 is the free-space permittivity constant, A_{atom}, is the area of one carbon atom, V_{app}, is the applied voltage and d is the distance between the atom and the trench bottom. The value used for the area of a carbon atom in graphene is 2.47678Å^2, which is determined by relaxing a periodically patterned sheet of graphene, then multiplying the relaxed length of the sheet by the relaxed width of the sheet and dividing by the number of carbon atoms it contains. The potential energy, U, stored in each parallel-plate capacitor unit is:

$$U = \frac{CV^2}{2} = \frac{QV}{2} = \frac{\varepsilon_0 A_{atom} V_{app}^2}{2d} \tag{13}$$

where U is calculated as a negative value in LAMMPS as it represents a potential well that pulls the ribbon into the trench bottom. The force, F, applied to each atom is calculated by taking the derivative of the potential energy with respect to the distance above the trench:

$$F = \frac{\mathrm{d}U}{\mathrm{d}d} = \frac{\varepsilon_0 A_{atom} V_{app}^2}{2d^2} \tag{14}$$

where a new value of F is applied to each atom every timestep as the individual atom's d value changes[46]. Once a graphene atom has moved within 1Å of the trench bottom, the simulation halts, as seen in Figure 18.

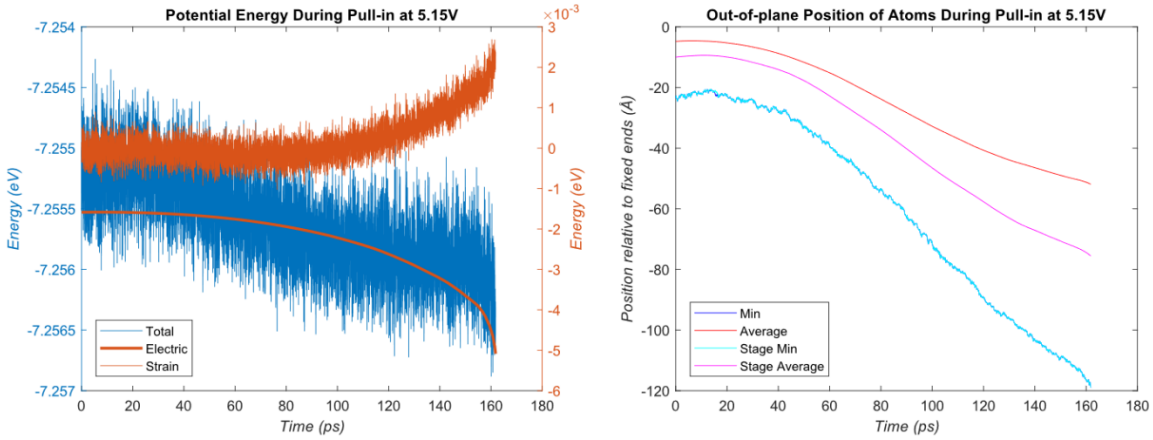

Figure 18: Per-atom energies and ribbon displacement during the simulation of a 1B ribbon under a 5.15V electrostatic load

To find V_p, the pull-in simulation is looped, where V_{app}, is decreased each iteration by the desired precision for the V_p result. When the ribbon fails to achieve the halting condition, the V_{app} corresponding to that iteration is called V_p. When voltages less than or equal to V_p are applied, the ribbon oscillates about a stable equilibrium position, as seen in Figure 19. The total potential energy in the system also oscillates as the strain energy and electrical potential energy alternate between their maximum and minimum values. In contrast, when a voltage above V_p is applied, the total potential energy enters a well where the electrical potential energy grows at a continually increasing rate, as seen in Figure 18. Results for each design's V_p can be seen in Table 8. Strain energy is calculated by subtracting the average equilibrium potential energy due to the carbon-carbon AIREBO bonds from the potential energy due to the bonds during pull-in.

Table 8: Trench depths and results for LAMMPS pull-in simulations

Design	**1B**	**2B**	**TS**
Trench Depth (Å)	120	150	25
V_p *(V)*	5.1	5.5	6.5

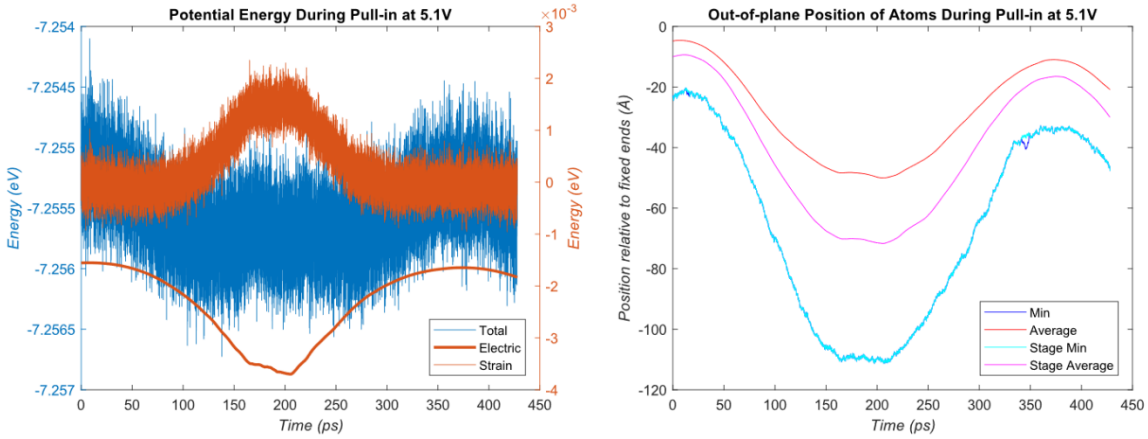

Figure 19: Per-atom energies and ribbon displacement during the simulation of a 1B ribbon under a 5.1V electrostatic load

4.3 Stress and Reaction Force Analysis

Under the pull-in conditions described above, fracture does not occur in any of the designs. Stresses, however, indicate that fracture is nearly achieved. During pull-in simulation, stress values for each atom are output every 100 timesteps. LAMMPS calculates a per-atom stress tensor, S, for individual atoms in the mobile graphene group in units of stress multiplied by volume:

$$S_{ab} = -\left[mv_a v_b + \frac{1}{2}\sum_{n=1}^{N_p}\left(r_{1_a}F_{1_b} + r_{2_a}F_{2_b}\right)\right] \tag{15}$$

where a and b cycle through x, y and z to form all six components of the stress tensor, m is the mass, v is the velocity, N_p is the number of pairwise interactions with neighboring atoms, r is the position of atoms in the pairwise interaction and F is the force on that atom resulting from the

pairwise interaction. Each component of the tensor can be divided by the per-atom volume, V_{atom}, to compute a per-atom stress tensor in units of stress:

$$\sigma_{ab} = \frac{S_{ab}}{V_{atom}} = \frac{S_{ab}}{A_{atom} t_{graphene}} \tag{16}$$

where the A_{atom} is calculated as described above and the thickness of the graphene nanoribbon, $t_{graphene}$, is 3.4Å. The von mises, σ_v, is used as the failure criterion:

$$\sigma_v = \sqrt{\frac{1}{2}\left[\left(\sigma_{xx} - \sigma_{yy}\right)^2 + \left(\sigma_{yy} - \sigma_{zz}\right)^2 + (\sigma_{zz} - \sigma_{xx})^2 + 6\left(\sigma_{xy}^2 + \sigma_{yz}^2 + \sigma_{xz}^2\right)\right]} \tag{17}$$

Figure 20 shows the von Mises stress distribution in each ribbon as contact is made during a pull-in simulation where V_{app} is equal to 0.1V more than V_p for each design.

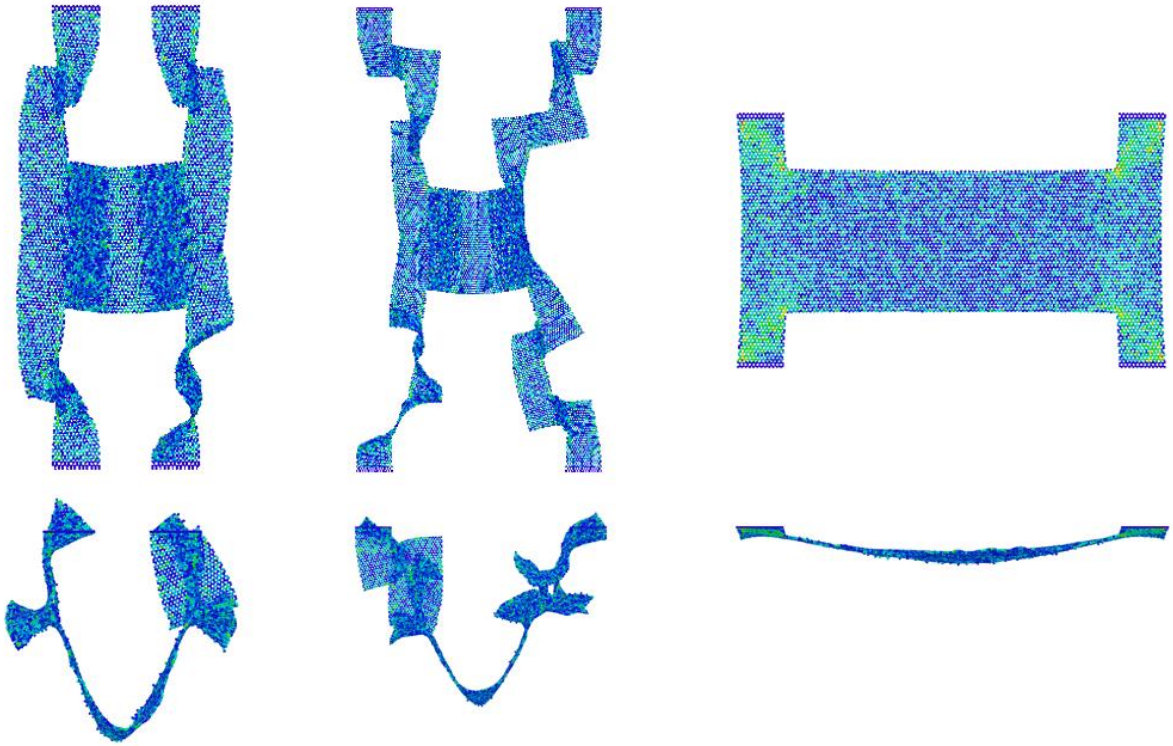

Figure 20: Top and front views of the 1B (left), 2B (middle) and TS (right) ribbons. Stress values are distributed between 100GPa (red) and below 10GPa (blue).

When averaged over time, the 1B and 2B ribbons show a nearly evenly distributed stress close to 20GPa. The TS ribbon shows stresses around 50GPa along the diagonals of the flexures with stresses approaching 100GPa near the corners. It is interesting to note that the von Mises stress is dependent on the rate of deformation during a dynamic analysis, especially in the 1B and 2B ribbons. When a load of 100V is applied, deformation occurs much more rapidly, and higher stresses are observed along the flexures, approaching fracture. Figure 21 shows the stress distribution in a 1B ribbon at contact with the trench when 100V is applied.

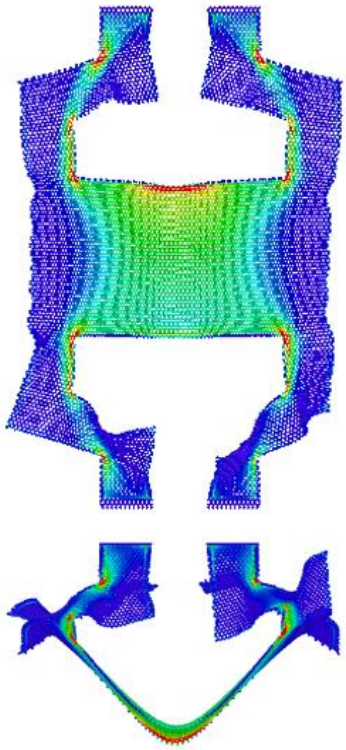

Figure 21: Top and front views of the stress distribution on a 1B ribbon under high voltage. Stress values are distributed between 100GPa (red) and below 10GPa (blue).

4.4 Comparison to Finite Element Analysis

To validate how well ANSYS models a graphene nanoribbon under electrostatic pull-in loads, a finite element analysis was repeated at the same dimensions of the LAMMPS simulations. The same setup was used as described in Chapter 3, but with the following differences: no PMMA layer was added above the graphene and material properties for pristine SCG were used. Results of each ribbon's equilibrium position as voltage is increased up to pull-in can be seen in Figure 22. The 1B, 2B and TS ribbons experience pull-in at 12.5V, 11.3V and 6.7V respectively. Note that the trench depths used are still 120Å, 150Å and 25Å.

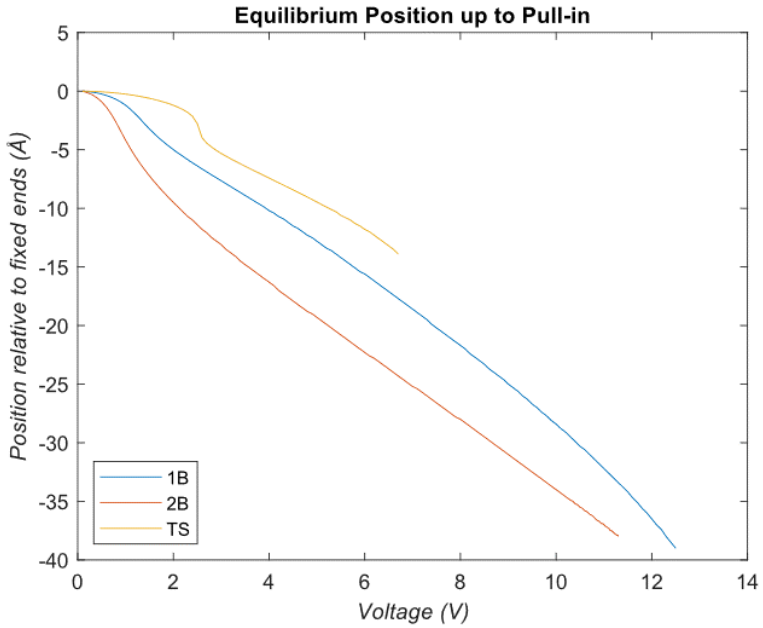

Figure 22: Ribbon equilibrium positions under increasing voltages

While the TS ribbon's finite-element V_p results agree closely with its molecular dynamics results, the 1B and 2B ribbons' pull-in results are about double those calculated by LAMMPS. This discrepancy suggests that a shell element is most appropriate where the ribbon geometry sustains more tension throughout the ribbon and avoids underconstraint. The finite element results reported here appear to deviate when large ripples appear during dynamic equilibrium resulting from low tension and underconstraint.

Dynamic FEA results agree with the above static FEA results. Figure 23 shows the displacement of a 1B ribbon over time. Oscillation at 11V indicates a stable equilibrium, while simulation failure during the 13V simulation indicates a voltage above V_p. The response time under a 13V load in ANSYS is just under 40ps, while the response time at 5.15V in the LAMMPS dynamic simulations is about 160ps.

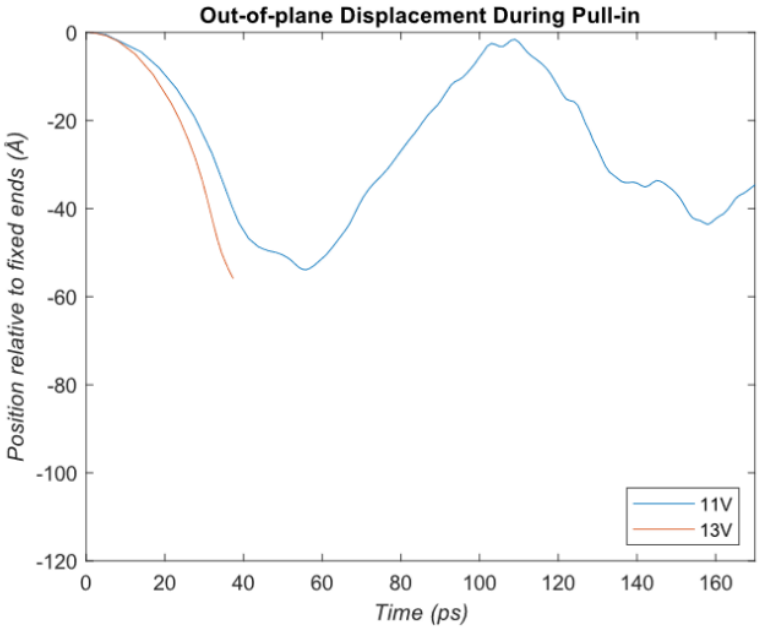

Figure 23: Results of a dynamic FEA simulation of a 1B ribbon at 13V and 11V

As was done in Chapter 3, a separate finite element analysis was performed to study stresses at pull-in. The TS ribbon's results are in better agreement than the other designs', again suggesting that the production of large ripples affects the accuracy of a shell element compared to a MD simulation. Furthermore, the 1B and 2B FEA results seen in Figure 24 reflect the MD results at high voltages. These results suggest that a high rate of displacement does not allow time for ripples to propagate, causing a smoother surface that can be described by the FEA methods used.

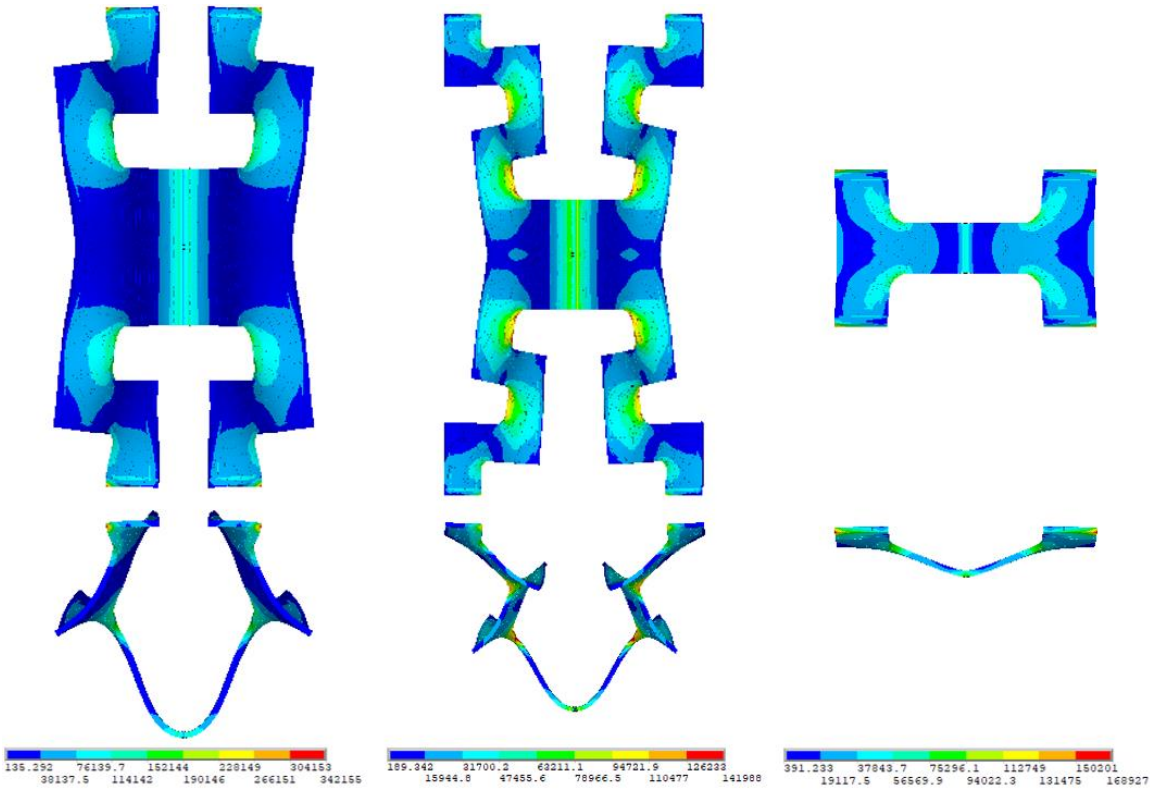

Figure 24: Stress distribution in at contact with trench bottom. All units are in MPa.

Chapter 5

Fabrication and Testing

This chapter is included as a means of evaluating the effectiveness of the above design and analysis methods. Fabrication and testing of the graphene nanoribbon ESD devices were performed by Jimmy Ng of UCLA's Semi-Conductor Materials Research Lab along with collaborating researchers at UCLA and UC Riverside. Figure 25 through Figure 29 are provided courtesy of Jimmy Ng, along with the data used to generate the reported values seen in Table 9. Several sizes of each device were fabricated and were studied in related research. This research focuses on the devices of the dimensions listed in Chapter 2.

5.1 Fabrication Technique

The graphene for each device was fabricated using a low-pressure chemical vapor deposition (LPCVD) process. As shown in Figure 25, carbon atoms are deposited on a copper foil in the presence of methane and hydrogen gasses in a vacuum chamber. This self-limiting process forms a single layer of polycrystalline graphene.

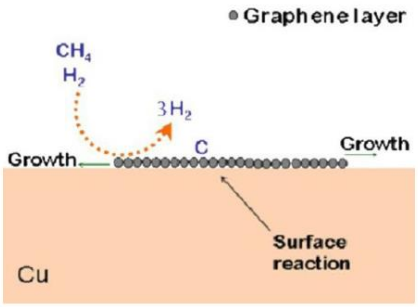

Figure 25: Self-limiting deposition of carbon atoms onto a copper foil

Once formed, the graphene can be transferred onto another substrate by depositing a layer of PMMA over the graphene, shown in Figure 26. The copper foil is then dissolved from the graphene which is then rinsed in DI water. The graphene is then placed on the target substrate, after which the PMMA layer can be removed using acetone.

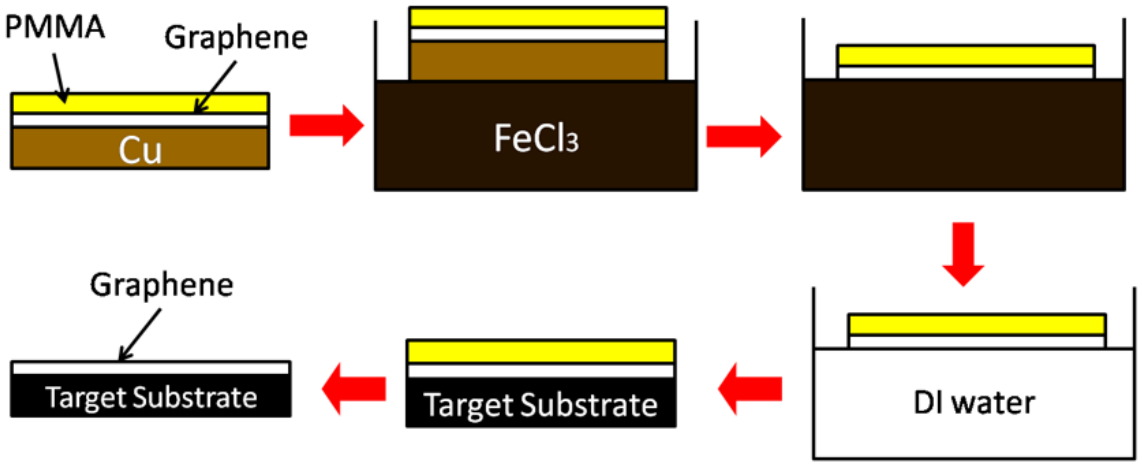

Figure 26: Process flow showing the transfer of graphene onto a substrate.

To fabricate the ESD devices, a doped silicon substrate was prepared with a layer of silicon dioxide and a layer of silicon nitride as shown in Figure 27(a),(b), and (c). The silicon nitride was etched in the location where each graphene nanoribbon would be etched, and the silicon dioxide was etched in preparation for a connection to ground through the silicon substrate. The graphene, prepared as described above, was then deposited and subsequently etched in the 1B, 2B and TS patterns determined by the mechanical design process. A network of metal pads was then deposited to allow the flow of electricity into the graphene and through the silicon ground. Finally, probes were connected to test the voltage and current[47].

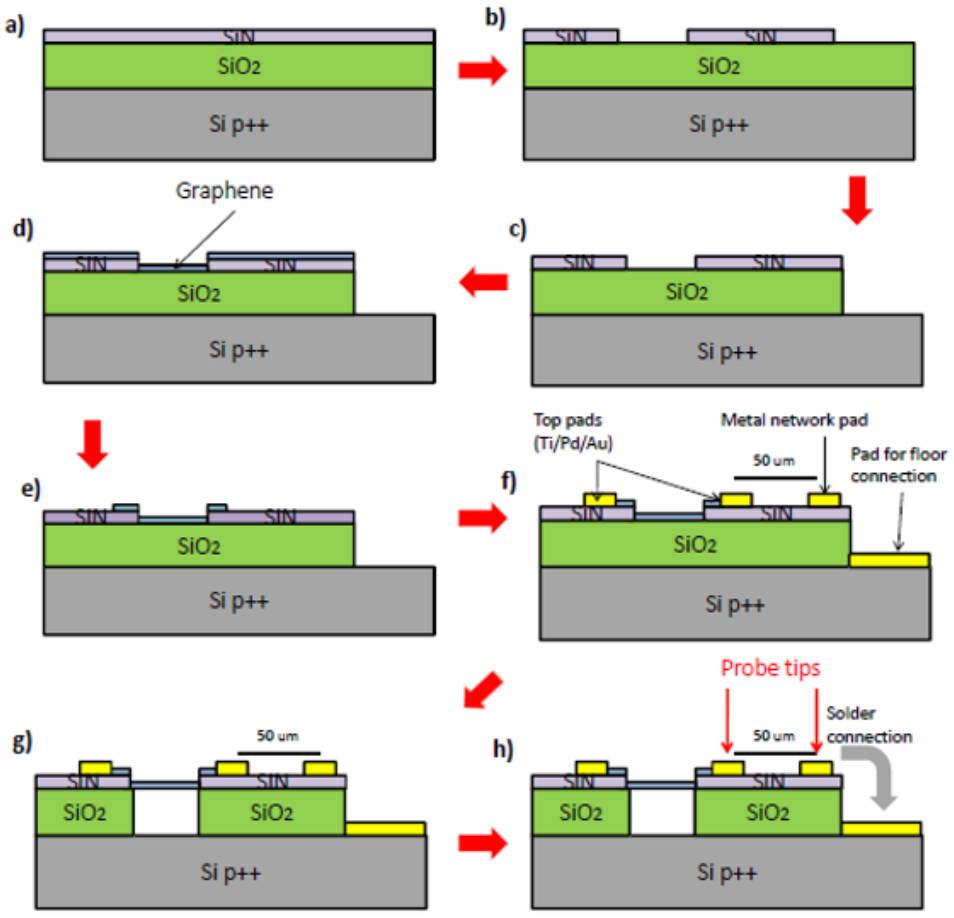

Figure 27: Process flow showing the preparation of the substrate for the graphene nanoribbon and the setup for testing of the graphene nanoribbon under ESD conditions.

5.2 Testing of Fabricated Devices and Experimental Results

Sample SEM images of the 1B, 2B and TS designs appear in Figure 28. As seen in the images, a small radius of about 1μm naturally forms on the interior and exterior corners of the ribbon patterns, a result of the etching process which does not appear in the patterns. These radii provide some stress relief against brittle failure during pull-in. Evidence of an error in the mask that produced the trench can also be seen in all three designs. The effect of this error is a lengthening of the flexures on one side of each design, which would theoretically result in lower stiffness and a lower pull-in voltage. In the case of the TS design, the offset results in a change in

geometry where the flexures connect back into a solid ribbon before reaching the clamped end conditions. Also visible in the TS SEM are the circular metal "nails" within the metal pad.

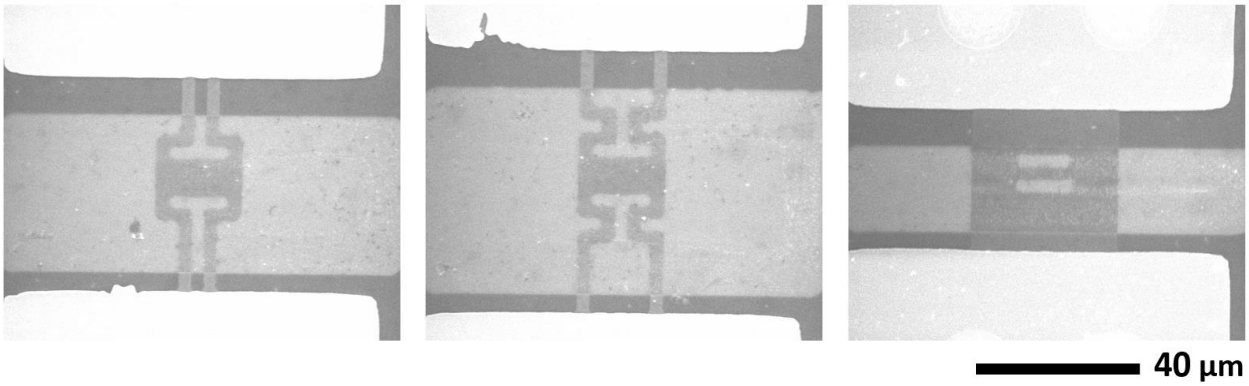

Figure 28: SEM images of the 1B (left), 2B (center) and TS (right). Appearing in the images are (from darkest to lightest) the silicon nitride layer, the graphene nanoribbon ESD lamina emergent mechanism, the doped silicon substrate and the metal pads.

The pull-in voltage for each design was experimentally determined by applying a voltage bias across the metal pads connected to the graphene and to the substrate, shown in Figure 27(h). This voltage was then slowly increased until the measured current sharply increased, indicating that the graphene had been pulled into the silicon substrate, creating a connection to ground that allows high-current electrical discharge, as shown in Figure 29.

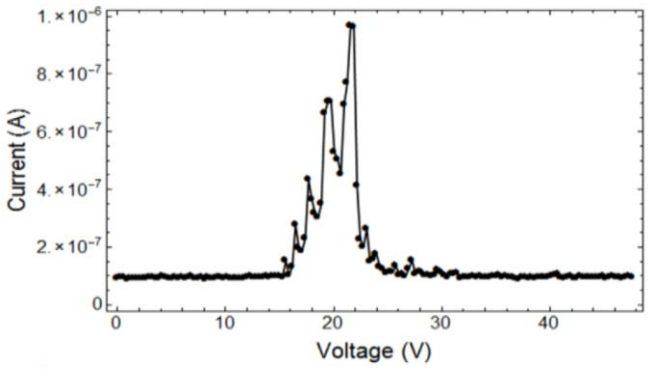

Figure 29: Typical I-V curve of a device during testing

The voltage at which the current began to flow was considered the pull-in voltage. As the voltage was increased past the pull-in voltage, thermal breakdown of the graphene ribbon occurred, and the current returned to zero. Results for each design can be seen in Table 9.

Only four 2B devices and three 1B devices were successfully fabricated and tested. The reported pull-in voltage is the average of the pull-in voltages of the successfully fabricated devices of each design. The 2B and 1B pull-in voltages are nearly two orders of magnitude higher than expected. The TS design consistently exhibited breakdown between 100V and 150V, which is also higher than expected. However, the breakdown mechanism for the TS design is likely due to a vapor arc process rather than pull-in and the previously described thermal breakdown of the graphene. Under a high electric field, the charged graphene could vaporize, allowing current to arc across the small air gap to ground before reaching a voltage where pull-in occurs[48]. It follows that the pull-in voltage of the TS design is higher than 150V but cannot be measured due to the vapor arc breakdown that occurs at a lower voltage.

Table 9: Experimentally determined pull-in voltages for each design. The 2B results are reported as the average of four successfully fabricated devices. The 1B results are reported as the average of three successfully fabricated devices.

Design	**1B**	**2B**	**TS**
Average Pull-in Voltage (V)	24.4	19.7	Not Measurable
Standard Deviation (V)	5.7	2.7	-

Despite the measured pull-in voltages being much higher than predicted, they still confirm the predicted trend that the 2B design would be the most flexible, allowing a lower pull-in voltage, and the TS design would be the stiffest, requiring a higher voltage for pull-in.

The devices also failed to demonstrate a repeatable pull-in behavior. When the voltage bias was removed, the 1B and 2B devices would demonstrate resistor-like behavior. Two possible explanations for this behavior exist: (1) similarly to the old design, the graphene ribbon or the PMMA layer fractured during pull-in, resulting in a broken ribbon that could not spring back up, or (2) the graphene ribbons were so flexible that they were not stiff enough to overcome the Van der Waals forces between the graphene and the silicon substrate, causing the ribbon to stick in the pulled-in position. Further testing supported the second explanation. Instead of using acetone for PMMA removal, a chloroform solution was substituted, resulting in a smaller residual PMMA layer. When these chloroform-treated devices were tested, they exhibited resistor-like behavior immediately, indicating that the graphene was already in contact with the substrate. Such behavior would result from a lowering of the ribbons' stiffness to the point where the Van der Waals forces could overcome the strain forces in the ribbon. As demonstrated in the FEA analyses, complete or partial removal of the PMMA layer would lower the device's stiffness. If the partial removal of PMMA from the graphene provides enough flexibility to be pulled in by Van der Waals forces, then a ribbon with a thick PMMA layer would also not overcome the Van der Waals after contact with the trench bottom.

Chapter 6

Conclusions

6.1 Discussion of New Designs and Modeling Techniques

Based on a beam theory approach, three new designs for a single-layer graphene nanoribbon ESD device are proposed and ready for fabrication and testing. All three can be fabricated by the same CVD and etching processes used for previous designs. Based on the models used in Chapters 2, 3 and 4 each design is expected to improve the repeatability of the ESD process. The 1B and 2B designs are expected to offer significant improvement in terms of slipping and material failure, while the TS design is expected to offer slight improvements. Improved repeatability is achieved by adding flexures into the design. These flexures bear the strain as the device is actuated, reducing stress at the ends of and throughout the graphene ribbon and the residual PMMA layer. Repeatability is anticipated to come at the cost of a low spring constant, however, and the 1B and 2B designs are not expected to achieve the design requirements for pull-in voltage under the size limitations available in this study.

Although the beam-theory equations and SDOF spring-mass model are limited in their ability to model graphene's mechanical behavior, they guide the design process well enough to improve the ribbon geometry in terms of increasing or decreasing its pull-in voltage and response time.

Finite element analyses in this study are based on nonlinear geometry and a MDOF model, which gives more accurate results than the SDOF beam model. They also provide a better way of incorporating residual layers of PMMA on the graphene ribbons. Using FEA, more information

on failure due to fracture and slipping can also be obtained. The experimentally determined V_p for the new designs, however, was about 100 times higher than the FEA predicted. Despite the discrepancy, the FEA model still predicts which designs will have higher pull-in voltages relative to each other and it does not rely on an experimentally determined fit factor.

On a smaller scale, MD simulations agree more closely with the FEA pull-in behavior in terms of voltage, response time and stresses. MD simulations are generally regarded as accurate when simulating graphene's mechanical properties, and they should even capture the effects of thermal rippling. It follows that the inaccuracy of the FEA on the larger scale is possibly due to any combination of the following: (1) the MD results and FEA results scale differently from each other when brought up to micrometer dimensions, (2) the FEA methods are appropriate for modeling SCG, but not PCG with a PMMA layer, (3) the thermal rippling seen in graphene nanoribbons cannot be adequately described by the FEA methods, and this inadequacy is accentuated in larger scale models, (4) the parallel-plate capacitor model used in both the FEA and MD simulations does not accurately describe the charge distribution in a graphene ribbon that occurs experimentally, (5) unaccounted for modification of graphene's electrical properties occurs in the presence of the PMMA layer. Of these reasons (3), (4) and (5) are most likely to result in the dramatically higher pull-in voltages from testing. Ripples in graphene that are improperly modeled could lead to higher out-of-plane stiffnesses. Nonclassical capacitance effects may exist at this device's dimensions and doping caused by the PMMA may further modify how a graphene ribbon responds to applied voltage.

6.2 Future Work

Future work can include optimizing these designs for the desired pull-in voltage of 1.5V under smaller feature size limitations. To achieve this, the discrepancies in pull-in voltage between the fabricated devices and the modeled devices must first be resolved. One way to resolve the discrepancies is to create a specific experimental constant as was done in the initial research that this work builds off. This strategy would not likely be scalable, however, and would not provide the desired precision as it would not eliminate the deviation seen in the tested results. A better analytical model could be developed to provide the needed accuracy while preserving the ease of optimization seen in the methods from Chapter 2. One possible route for creating such a model would be to use a modified FACT method that accounts for nonlinear geometry. This approach would provide the framework of a well-developed design process and extend the model into multiple degrees of freedom.

A van der Waals force implementation for the finite element approach could also improve the model. This would provide more insight into the device's failure to spring back. In conjunction with this model, a new batch of chloroform-treated devices could be fabricated and tested. This batch of devices could be studied more thoroughly to determine PMMA thickness. If the amount of PMMA is significantly reduced, the pull-in voltage of these devices would address the question of PMMA's effect on the nanoribbons' electrical behavior.

To study the effect of the thermal rippling on pull-in voltage, new designs could be created that avoid underconstraint. While the underconstraint allows the device to displace more with lower stresses, it also allows for poor dynamic behavior, which is shown in the MD simulations to be especially complicated due to graphene's thermal ripples.

The material properties used in the FEA approach can possibly be modified to improve accuracy. If additional stiffness due to thermal ripples can be characterized, that could be implemented. The stress-strain curve exhibited by graphene in tension is also very nonlinear. A hyperelastic model exhibiting an elasticity that follows this curve might be more suitable than using just the material's Young's modulus[49]. The SCG model being compared to MD simulations could also be modeled as an anisotropic material with different stiffnesses in its armchair and zigzag orientations[37].

Development of analytical models suitable for graphene's mechanical behavior as a 2D material will be another important advance in the future. The development of these models could involve the use of molecular dynamics to study how graphene's thermal rippling affects its mechanical performance.

Where sufficient computational resources are available, an MD simulation could be performed on the actual scale of the fabricated devices. This would address the questions of how scalable the MD simulation is but would also require a large amount of time. Other research has successfully modeled PMMA using LAMMPS. The addition of PMMA to an MD simulation would also significantly increase the computational cost.

Once a refined model for the mechanical behavior of graphene nanoribbons has been developed, work can be done on streamlining a design process for devices such as the ESD device in this work.

Appendix A

MATLAB Scripts

This appendix includes the MATLAB functions used with the SDOF model from Chapter 2 to calculate pull-in voltages for each of the designs, run a dynamic simulation of pull-in and set the optimizer settings.

A1. SDOF Beam Model Spring Constant Calculator (k_calculator.m)

This function calculates the SDOF beam model spring constant for a given ribbon design. This spring constant can then be used to calculate the pull-in voltage and natural frequency for the simplified approach. It must be called by a script that specifies the geometric parameters and an experimental fit factor as described in Chapter 2.

```
function k_eff = k_calculator(design,C_nonpristine,E,G,h,L,b,t,e)
% Calculate the k_eff value based on the selected design

switch design
    case '2blade'
        n_series = 2; %number of blade elements in series
        n_parallel = 4; %number of parallel spring structures
            k_leg = k_castigliano(E,G,h,L,b,t,e,n_series);
            k_eff = n_parallel * k_leg;
    case '1blade'
        n_series = 1; %number of blade elements in series
        n_parallel = 4; %number of parallel spring structures
            k_leg = k_castigliano(E,G,h,L,b,t,e,n_series);
            k_eff = n_parallel * k_leg;
    case 'oldRibbon'
        I = b*h^3/12;
        k_eff = 384*E*I/L^3; %fixed-fixed distr load model
    case 'TS'
        I = b*h^3/12;
        k_eff = 384*E*I/(5*L^3); %simply supp distr load model
end

% Multiply by an experimentally determined correction factor for non-Pristine
graphene cases
k_eff = k_eff*C_nonpristine;
end
```

A2. Castigliano Spring Constant Calculator (k_castigliano.m)

This function uses Castigliano theorem to calculate the spring constant of a single leg of a 1B or 2B ribbon. It is written to be called by the above 'k_calculator.m' function.

```
function k_leg = k_castigliano(E,h,L,b,t,e,n_series)

% F = k_leg*y
% k_leg = F/y = 1/c
% OR y/F = c = 1/k_leg

% find c and then invert
% c = sum of the c for each member

G = 280e9; %Pa

if n_series == 1
    L1 = 0;
    L2 = 0;
    L3 = e+b/2;
    L4 = t/2 + L + t/2;
    L5 = b/2 + e;
elseif n_series == 2
    L1 = e + b/2;
    L2 = t/2 + L + t/2;
    L3 = b/2 + e + b/2;
    L4 = t/2 + L + t/2;
    L5 = b/2 + e;
end

I1 = t*h^3/12;
I2 = b*h^3/12;
I3 = t*h^3/12;
I4 = b*h^3/12;
I5 = t*h^3/12;

J1 = PolarMomInertia(t,h);
J2 = PolarMomInertia(b,h);
J3 = PolarMomInertia(t,h);
J4 = PolarMomInertia(b,h);
J5 = PolarMomInertia(t,h);

k_shear = 1.5;
A1 = h * t;
A2 = h * b;
A3 = h * t;
A4 = h * b;
A5 = h * t;

c1 = (L1^3/3 + L1^2*L3 + L1^2*L5 + L1*L3^2 + L1*L5^2 + 2*L1*L3*L5)/(E*I1) +
((L4-L2)^2*L1)/(G*J1) + (k_shear*L1)/(G*A1);
c2 = (L2^3/3 - L2^2*L4 + L2*L4^2)/(E*I2) + ((L5+L3)^2*L2)/(G*J2) +
(k_shear*L2)/(G*A2);
```

```
c3 = (L3^3/3 + L3^2*L5 + L5^2*L3)/(E*I3) + (L4^2*L3)/(G*J3) +
(k_shear*L3)/(G*A3);
c4 = (L4^3)/(3*E*I4) + (L5^2*L4)/(G*J4) + (k_shear*L4)/(G*A4);
c5 = (L5^3)/(3*E*I5) + (k_shear*L5)/(G*A5);

% k1 = 1/c1;
% k2 = 1/c2;
% k3 = 1/c3;
% k4 = 1/c4;
% k5 = 1/c5;

c_leg = c1 + c2 + c3 + c4 + c5;
k_leg = 1/c_leg;
end
```

A3. Polar Moment of Inertia Calculator (PolarMomInertia.m)

This function calculates the polar moment of inertia for use in the Castigliano theorem.

```
function J = PolarMomInertia(W,T)

Temp=0;
    if(W>T)
        for n=1:2:100
            Temp=Temp+(tanh(n*pi*W/(2*T))/(n^5));
        end
        J=((T^3)*W/3)*(1-((192*T/((pi^5)*W))*Temp));
    else
        for n=1:2:100
            Temp=Temp+(tanh(n*pi*T/(2*W))/(n^5));
        end
        J=((W^3)*T/3)*(1-((192*W/((pi^5)*T))*Temp));
    end
```

A4. SDOF Dynamic Pull-in Simulator (pullin.m)

This function simulates and plots the dynamic response of a SDOF graphene ribbon model under an applied voltage up until contact with the trench bottom. This function can be called iteratively to find the pull-in voltage. More details are found in the comments within the code.

```
%-------------------------------------------------------------------------
% This function takes in trench, spring and applied static voltage
% parameters and outputs the time it takes for the discharge stage
% to be pulled in to the trench bottom
%-------------------------------------------------------------------------

function t_contact = pullin(d,m,k_eff,damp,A,V_app,optimize,linestyle)
% Constants
```

```
eps_0 = 8.85*10^-12;    %Vacuum premissivity, F/m

% ODE setup
x0_pullin = [0;0]; %initial conditions for pullin [position,velocity]
t_span_pullin = [0 2e-5]; %time span for mechanical pullin response simulation

% setup for Pullin Contact Event
dischargeContactParams = @(t,x) dischargeContact(t,x,d); %define function that
carries the trench depth parameter into the contact event funciton
options = odeset('Events',dischargeContactParams); %use event function to
indicate when stage makes contact with trench bottom

% Run ODE
[t_pullin,x_pullin,t_contact,x_contact,i_contact]        =        ode45(@(t,x)
MechResp(t,x,m,k_eff,damp,eps_0,A,V_app,d),t_span_pullin,x0_pullin,options);

if optimize == 0
    % Plot the platform's position vs time
    semilogx(t_pullin,x_pullin(:,1)*10^6,linestyle);
    hold on
    ttl = sprintf('Pull-in Response');
    xlim([1e-10 2e-5])
    %title(ttl)
    xlabel('Time (s)')
    ylabel('Position (\mum)')
end
end
```

A5. SDOF Equations of Motion (MechResp.m)

This function is written to be called by the ode45 function in 'pullin.m'. It contains the equations of motion that govern the behavior of the SDOF ribbon model.

```
%-------------------------------------------------------------------------
% This function is for use with ode45 to output the position and velocity
% of the discharge stage given flexure properties, stage properties, and
% electrical properties
%-------------------------------------------------------------------------

function dxdt = MechResp(t,x,m,k,b,eps_0,A,V_app,d)

% differential equation setup - here we calculate the time derivatives of
% the discharge stage's position, x
% BRING IN intial position and velocity, other parameters, time range
% OUTPUT x(1) = x OR position, x(2) = x' OR velocity
% CALC dxdt(1) = x' OR velocity, dxdt(2) = x'' OR acceleration

dxdt = zeros(2,1);
dxdt(1) = x(2);
dxdt(2) = (1/m)*(-k*x(1)-b*x(2)+(eps_0*A*V_app^2)/(2*(d-x(1))^2));
```

A5. Pull-in Contact Trigger (dischargeContact.m)

This function is written to be called by the ode45 function in 'pullin.m' to indicate pull-in and terminate the simulation.

```
%-------------------------------------------------------------------------
% This function defines the event for stopping MechResp, indicating contact
% at the bottom of the trench and electrostatic discharge
%-------------------------------------------------------------------------

function [value,isterminal,direction] = dischargeContact(t,x,d)
value = .9999*d-x(1); % depth  - displacement
isterminal = 1; % stop the integration
direction = 0; % any direction
end
```

A6. Optimization Settings (optimizer.m)

This function sets the settings for use with the MATLAB optimizer. It must be called by a function that specifies a vector the initial guesses of the parameters to be optimized ($x0$), bounds for those parameters (lb, ub) and other design conditions for those parameters ($Aineq, bineq, Aeq, Beq$). 'OptimizableDesign' must be a function that takes the parameters to be optimized, which are $A, d\ and\ k,$ as an input and outputs the quantity to be minimized, which is $1/V_p^2$ as described in Chapter 2.

```
function            [x,fval,exitflag,output,lambda,grad,hessian]            =
optimizer(x0,Aineq,bineq,Aeq,Beq,lb,ub)

%% Start with the default options
options = optimoptions('fmincon');

%% Modify options setting
options = optimoptions(options,'Display', 'off');
options = optimoptions(options,'OptimalityTolerance', 1e-12);
options = optimoptions(options,'FunctionTolerance', 1e-12);
options = optimoptions(options,'StepTolerance', 1e-12);
options = optimoptions(options,'FunValCheck', 'on');
options = optimoptions(options,'PlotFcn', {  @optimplotx @optimplotfunccount
@optimplotfval          @optimplotconstrviolation          @optimplotstepsize
@optimplotfirstorderopt });
options = optimoptions(options,'ConstraintTolerance', 1e-12);
%options = optimoptions(options,'OutputFcn', @outfun);
[x,fval,exitflag,output,lambda,grad,hessian] = ...
```

```
fmincon(@OptimizableDesign,x0,Aineq,bineq,Aeq,Beq,lb,ub,[],options);
```

Appendix B

ANSYS Scripts

The following scripts were used with ANSYS Mechanical APDL to determine the pull-in voltage and stresses that correspond to each design. Included here are scripts for 1B designs.

B1. Pull-in Voltage Determination Using Voltage Load Steps

This script runs a static simulation of a PCG graphene-PMMA composite ribbon at the size specified for fabrication.

```
FINISH
/FILNAME,ANSYS_1B_comp,1

FINISH
/CLEAR

SAVE, ANSYS_1B_comp, ,

      /title, Electrostatic-Structural  Direct Analysis

FINISH
/BEGIN

      ! Parameters (uMKSV unit system)
            !Mesh
                  elmsize = .5      !um

            !Geometry
                  h =.34e-3       ! beam height, um
                  hp = 4.12e-3
                  l = 21      ! length of beam, um
                  w = 9           ! beam width, um
                  l_1 = 3
                  w_1 = 3
                  l_2 = 3
                  w_2 = 3

                        w_tot = w/2+l_1+w_2+l_1
                  d = 1.000         ! gap, um

            !Material
                  E = 600e3*nonPFactor    !MPa [(kg)/(um)(s2)]
                  nu = .165
```

```
                dens = 2.266e-15        !kg/um3

                Ep = 2450
                nup = 0.35
                densp = 1185*1e-18

          !Electric
                V = 15                              ! applied voltage, V
                V_int = .001
                epse=1                    ! air permittivity, relative

FINISH
/PREP7

     !Element Types
          et,1,shell281,,          ! Shell element for graphene ribbon !<---
-----------------------------------------
          et,2,SOLID226,1001,,,1   ! 20-node "elastic air" brick
          et,3,shell281,,          ! Shell element for PMMA layer !

     !Material Properties
          !Graphene
                mp,ex,1,E            ! MPa
                mp,nuxy,1,nu
                mp,dens,1,dens      ! kg/(um)^3

          !Air
                mp,ex,2,1.e-6            ! MPa
                mp,prxy,2,0.0
                mp,perx,2,1
                emunit,EPZRO,8.854e-6          !   pF/um   (define   FreeSpace
Permitivity with proper units)

          !PMMA
                mp,ex,3,Ep            ! MPa
                mp,nuxy,3,nup
                mp,dens,3,densp      ! kg/(um)^3

     !Geometry
          !Sections
                !Graphene ribbon section !secdata, TK, MAT, THETA, NUMPT,
LayerName
                      sectype,1,shell,
                      secdata,h,1,0,9
                      secdata,hp,3,0,9
                      secoffset,BOT

          !Volumes
                block,0,l/2,-d,0,w/2,0                    !Discharge stage
                block,l/2,l/2-w_1,-d,0,w/2,w/2+l_1  !torsion length 1
                block,l/2,l/2-(w_1+l_2+w_1),-d,0,w/2+l_1,w/2+l_1+w_2
!bending length
                block,l/2-(w_1+l_2+w_1),l/2-(w_1+l_2+w_1)+w_1,-
d,0,w/2+l_1+w_2,w/2+l_1+w_2+l_1 !torsion length 2
                vglue,all
```

```
            !Mesh
                  !Discharge Structure/Ribbon
                        vsel,all                        !(select beam volume)
                        aslv                            !(select  areas  in  the
volume)
                        asel,r,loc,y,0
                        lsla                            !(select  lines  in  the
areas)
                        lesize,all,elmsize              !divide  all  lines  into
elements of specified size

                        type,1            !assign element type
                        !mat,1            !assign material type
                        amesh,all   !mesh areas

                        !view the area
                        /eshape,1.0
                        /EDGE,1,0,45
                        /GLINE,1,0
                        /replot

                  !Air Gap
                        msha,1,3d                ! set element shape to tet
                        mshmid,2                 ! drop mid-side nodes

                        vsel,all
                        aslv
                        lsla
                        lsel,r,loc,y,-d
                        lesize,all,elmsize         !divide  bottom  lines  into
elements of specified size
                        lsla
                        lsel,r,loc,y,-d/2
                        lesize,all,,,1          !divide   thickness   into   1
element

                        type,2
                        mat,2
                        vmesh,5,8

                        !view the area
                        /view,1,1,1,1
                        /number,1
                        /pnum,type,1
                        eplot

            !Boundary Conditions
                  !Make symmetric about origin X-Z
                        asel,s,loc,z,0
                        asel,a,loc,x,0
                        aplo
                        DA,all,SYMM

                  !structural BC on connected area
                        asel,s,loc,z,w_tot
                        aplot
```

```
                        DA, all, UX , 0
                        DA, all, UY , 0
                        DA, all, UZ , 0
                        DA, all, ROTX , 0
                        DA, all, ROTY , 0
                        DA, all, ROTZ , 0

                  !structural BC on bottom areas
                        asel,s,loc,y,-d
                        aplot
                        DA, all, UX , 0
                        DA, all, UY , 0
                        DA, all, UZ , 0

                  ! electrical BC on bottom areas
                        asel,s,loc,y,-d
                        aplot
                        da,all,volt,0           ! ground

                  ! electrical BC on top areas
                        asel,s,loc,y,0
                        aplot
                        da,all,volt,V           ! electrode

                  asel,all
                  aplo

FINISH
/SOLU
      !Convergence
            cnvtol,f,1,1e-6         !set tolerance on force convergence
            neqit,50                !set max number of equilibrium iterations

      !Time Steps
            time,V              ! Time = voltage
            deltim,V_int           !Voltage interval
            kbc,0               ! ramped loading

      !Output data
            outres,all,all

      !Nonlinear geometry
            nlgeom,on
      !Solution
            solve
            SAVE, ANSYS_1B_comp, ,



FINISH
/POST1  !------POST PROCESSING--------------------

      !load the last set of data
            SET,LAST

      !load the set of data previous to the current one
```

```
            SET,PREVIOUS

      !get the timestep for the current voltage's data
            *GET, V_PULLIN, ACTIVE, 0, SET, TIME


FINISH
/POST26

      !set the min and max times to plot and list out displ over voltage
            PLTIME, , V_PULLIN
            PRTIME, , V_PULLIN
            !Plot the displacement of nodes
                  resnode2 = node(0,0,0)    ! node for displacement display
                  resnode3 = node(0,0,w/2)    ! node for displacement display
                  resnode4 = node(l/2,0,0)    ! node for displacement display
                  resnode5 = node(l/2,0,w/2+w_1+w_2)    ! node for displacement
display
                  resnode6  =  node(l/2-2*l_1-l_2,0,w/2+w_1)        !  node  for
displacement display
                  nsol,2,resnode2,u,y,stage_center
                  nsol,3,resnode3,u,y,stage_edge_midlength
                  nsol,4,resnode4,u,y,stage_edge_midwidth
                  nsol,5,resnode5,u,y,flexure_inner
                  nsol,6,resnode6,u,y,flexure_outer

                  /OUTPUT, ANSYS_1B_comp_displ, txt, ,
                  prvar,2,3,4,5,6
                  /OUTPUT

                  /axlab,y,UY
                  /axlab,x,Voltage

                  plvar,2,3,4,5,6

                  !turn off csys
                  /PLOPTS,INFO,3
                  /PLOPTS,LEG1,1
                  /PLOPTS,LEG2,1
                  /PLOPTS,LEG3,1
                  /PLOPTS,FRAME,0
                  /PLOPTS,TITLE,1
                  /PLOPTS,MINM,0
                  /PLOPTS,FILE,0
                  /PLOPTS,SPNO,0
                  /PLOPTS,WINS,1
                  /PLOPTS,WP,0
                  /PLOPTS,DATE,2
                  /TRIAD,OFF
                  /REPLOT

                  !white background with black text
                  /RGB,INDEX,100,100,100, 0
                  /RGB,INDEX, 80, 80, 80,13
                  /RGB,INDEX, 60, 60, 60,14
                  /RGB,INDEX, 0, 0, 0,15
```

```
                /REPLOT

                /image,save,ANSYS_1B_comp_displ,png


FINISH
/POST1
      !Plot just the model
            !make smoother surfaces
            /EFACET,2
            PLDI, ,
                  !turn off csys
                  /PLOPTS,INFO,3
                  /PLOPTS,LEG1,1
                  /PLOPTS,LEG2,1
                  /PLOPTS,LEG3,1
                  /PLOPTS,FRAME,0
                  /PLOPTS,TITLE,1
                  /PLOPTS,MINM,0
                  /PLOPTS,FILE,0
                  /PLOPTS,SPNO,0
                  /PLOPTS,WINS,1
                  /PLOPTS,WP,0
                  /PLOPTS,DATE,2
                  /TRIAD,OFF
                  /REPLOT

      !Plot the symmetric results
            !EXPAND, Nrepeat, Type, Method, DX, DY, DZ
            /EXPAND,2,RECT,HALF,0.0001,,EXPAND,2,RECT,HALF,,,0.0001
            eplot

      !Plot at trimetric view
      /view,1,1,1,1
      /AUTO,1
      /replot

      !animate the deformed shape
            !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
      MIN,  MAX
            ANTIME,     20,         0.25, 1,          1,                2,
            0,          V_PULLIN
            !save the animation to a file
                  !/ANFILE, LAB, FNAME, EXT, DIR
                  /ANFILE,SAVE,ANSYS_1B_comp_DEFORM,AVI,

      ! Plot the displacement from 0 to vpullin
            PLNS,U,Y
            !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
      MIN,  MAX
            ANTIME,     20,         0.25, 1,          1,                2,
            0,          V_PULLIN
            !save the animation to a file
                  !/ANFILE, LAB, FNAME, EXT, DIR
                  /ANFILE,SAVE,ANSYS_1B_comp_DISPL,AVI,
      ! Plot the stress from 0 to vpullin
```

```
      PLNS,S,EQV
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_comp_VONMISES,AVI,

! Plot the forces from 0 to vpullin
      PLNS,FMAG,Z
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_comp_YForces,AVI,

SAVE, ANSYS_1B_comp, ,
```

B2. Stress Analysis Under Displacement at Contact

This script runs a static simulation of a PCG graphene-PMMA composite ribbon at the size specified for fabrication.

```
FINISH
/FILNAME,ANSYS_1B_cont_comp,1

FINISH
/CLEAR

SAVE, ANSYS_1B_cont_comp, ,

      /title, Structural Direct Analysis

FINISH
/BEGIN

      ! Parameters (uMKSV unit system)
            !Mesh
                  elmsize = .5      !um

            !Geometry
                  h =.34e-3       ! beam height, um
                  hp = 4.12e-3
                  l = 21      ! length of beam, um
                  w = 9           ! beam width, um (across the trench)
                  l_1 = 3
                  w_1 = 3
                  l_2 = 3
                  w_2 = 3
```

```
                  w_tot = w/2+l_1+w_2+l_1
            d = 1.000         ! gap, um
            fillet = 1
      !Material

            E = 600e3   !MPa [(kg)/(um)(s2)]
            nu = .165
            dens = 2.266e-15

            Ep = 2450
            nup = 0.35
            densp = 1185*1e-18

      !Load
            displ = d                     ! applied displacement, V
            displ_int = .01

FINISH
/PREP7

   !Element Types
      et,1,shell281,,          ! Shell element for graphene ribbon
      et,2,SOLID226,1001,,,1   ! 20-node "elastic air" brick
      et,3,shell281,,          ! Shell element for PMMA layer !

   !Material Properties
      !Graphene
            mp,ex,1,E            ! MPa
            mp,nuxy,1,nu
            mp,dens,1,dens      ! kg/(um)^3

      !Air
            mp,ex,2,1.e-6            ! MPa
            mp,prxy,2,0.0
            mp,perx,2,1
            emunit,EPZRO,8.854e-6          !  pF/um  (define  FreeSpace
Permitivity with proper units)

      !PMMA
            mp,ex,3,Ep            ! MPa
            mp,nuxy,3,nup
            mp,dens,3,densp      ! kg/(um)^3

   !Geometry
      !Sections
            !Graphene ribbon section !secdad, TK, MAT, THEd, NUMPT,
LayerName
                  sectype,1,shell,
                  secdad,h,1,0,9
                  secdad,hp,3,0,9
                  secoffset,BOT

      !Areas
            RECTNG,0,l/2,w/2,0                        !Discharge sdge
            RECTNG,l/2,l/2-w_1,w/2,w/2+l_1      !torsion length 1
```

```
                RECTNG,l/2,l/2-(w_1+l_2+w_1),w/2+l_1,w/2+l_1+w_2    !bending
length
                RECTNG,l/2-(w_1+l_2+w_1),l/2-
(w_1+l_2+w_1)+w_1,w/2+l_1+w_2,w/2+l_1+w_2+l_1 !torsion length 2
                aglue,all
                LFILLT,21,8,fillet, ,
                LFILLT,23,8,fillet, ,
                LFILLT,24,14,fillet, ,
                lsel,u,,,5
                lsel,u,,,6
                lsel,u,,,10
                lsel,u,,,11
                lsel,u,,,16
                lsel,u,,,25
                lsel,u,,,17
                lsel,u,,,7
                lsel,u,,,19
                lplot
                al,all

          !Mesh
                !Discharge Structure/Ribbon
                      lesize,all,elmsize            !divide  all  lines  into
elements of specified size
                      lesize,3,elmsize/2
                      lesize,9,elmsize/2
                      lesize,13,elmsize/2
                      type,1            !assign element type
                      mat,1       !assign material type
                      amesh,1     !mesh areas

                      !view the area
                      /eshape,1.0
                      /EDGE,1,0,45
                      /GLINE,1,0
                      /replot

          !Plot at trimetric view
                !/view,1,1,1,1


          !Boundary Conditions
                !Make symmetric about origin X-Z
                      nsel,s,loc,x,0
                      d,all,ROTY,0
                      d,all,UX,0
                      nsel,all
                      nsel,s,loc,y,0
                      d,all,ROTX,0
                      d,all,UY,0
                      nsel,all

                !structural BC on connected area
                      nsel,s,loc,y,w_tot
                      nplot
```

```
                        D, all, UX , 0
                        D, all, UY , 0
                        D, all, UZ , 0
                        D, all, ROTX , 0
                        D, all, ROTY , 0
                        D, all, ROTZ , 0


                  ! displacement held at bottom
                        nsel,s,loc,x,0,w/10
                        nplot
                        d,all,UZ,-displ           ! ground

                  nsel,all
                  nplo

FINISH
/SOLU
      !Convergence
            cnvtol,f,1,1e-6         !set tolerance on force convergence
            neqit,50                !set max number of equilibrium iterations

      !Time Steps
            time,displ              ! Time = displacement
            deltim,displ_int           !displacement interval
            kbc,0               ! ramped loading

      !Ouhput dad
            outres,all,all

      !Nonlinear geometry
            nlgeom,on

      !Solution
            solve
            SAVE, ANSYS_1B_cont_comp, ,

FINISH
/POST1 !POST Processing ----------

      !load the last set of dad
            SET,LAST

      !list reaction solution
            PRSOL

      !Plot just the model
            PLDI, ,
                  !turn off csys
                  /PLOPTS,INFO,3
                  /PLOPTS,LEG1,0
                  /PLOPTS,LEG2,1
                  /PLOPTS,LEG3,1
                  /PLOPTS,FRAME,0
                  /PLOPTS,TITLE,1
                  /PLOPTS,MINM,0
```

```
            /PLOPTS,FILE,0
            /PLOPTS,SPNO,0
            /PLOPTS,WINS,1
            /PLOPTS,WP,0
            /PLOPTS,DATE,0
            /TRIAD,OFF
            /REPLOT

!Plot the symmetric results
      !EXPAND, Nrepeat, Type, Method, DX, DY, DZ
      /EXPAND,2,RECT,HALF,0.0001,,,2,RECT,HALF,,0.0001,,,
      eplot

!Plot at trimetric view
      /view,1,1,1,1
      /AUTO,1

!make smoother surfaces
      /EFACET,4
      /GLINE,2,-1
      /GLINE,1,-1
      /REPLOT

PLNS,U,Z

!Change the contour legend range (use to show PMMA stresses)
      !/CONTOUR, WN, NCONT, VMIN, VINC, VMAX
      /contour,1,9,,,
      /replo

!animate the deformed shape
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_cont_comp_DEFORM,AVI,

! Plot the displacement from 0 to vpullin
      PLNS,U,Z
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_cont_comp_DISPL,AVI,

! Plot the stress from 0 to vpullin
      PLNS,S,EQV
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
```

```
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_cont_comp_VONMISES,AVI,

! Plot the forces from 0 to vpullin
      PLNS,FMAG,Y
      !ANTIME,NFRAM,    DELAY,      NCYCL,      AUTOCNTRKY, RSLTDAT,
MIN,  MAX
      ANTIME,     20,         0.25, 1,          1,                2,
      0,          V_PULLIN
      !save the animation to a file
            !/ANFILE, LAB, FNAME, EXT, DIR
            /ANFILE,SAVE,ANSYS_1B_cont_comp_YForces,AVI,

SAVE, ANSYS_1B_cont_comp, ,
```

B3. Dynamic Analysis

This script simulates an SCG graphene ribbon at the same dimensions used in the LAMMPS simulations.

```
FINISH
/FILNAME,ANSYS_1B_dyn_v13,1

FINISH
/CLEAR

SAVE, ANSYS_1B_dyn_v13, ,

      /title, Electrostatic-Structural ANSYS_1B_dyn_v13

FINISH
/BEGIN

      ! Parameters (uMKSV unit system)
            !Mesh
                  elmsize = .5e-3   !um

            !Geometry
                  h =.34e-3       ! beam height, um
                  l = 210e-4        ! length of beam, um
                  w = 90e-4           ! beam width, um
                  l_1 = 30e-4
                  w_1 = 30e-4
                  l_2 = 30e-4
                  w_2 = 30e-4

                        w_tot = w/2+l_1+w_2+l_1
                  d = 120e-4         ! gap, um

            !Material
                  E = 1000e3  !MPa [(kg)/(um)(s2)]
                  nu = .165
```

```
                dens = 2.266e-15

            !Electric
                V = 13                              ! applied voltage, V
                epse = 1                    ! air permittivity, relative
                time_int = 1e-12 !sec

FINISH
/PREP7

      !Element Types
            et,1,shell281,,          ! Shell element for graphene ribbon !<---
-----------------------------------------
            et,2,SOLID226,1001,,,1   ! 20-node "elastic air" brick

      !Material Properties
            !Graphene
                mp,ex,1,E             ! MPa
                mp,nuxy,1,nu
                mp,dens,1,dens      ! kg/(um)^3

            !Air
                mp,ex,2,1.e-16            ! MPa
                mp,prxy,2,0.0
                mp,perx,2,1
                emunit,EPZRO,8.854e-6          !  pF/um  (define  FreeSpace
Permitivity with proper units)

      !Geometry
            !Sections
                !Graphene ribbon section
                    sectype,1,shell,
                    secdata,h,1,0,9
                    secoffset,BOT
            !Volumes
                block,0,l/2,-d,0,w/2,0                    !Discharge stage
                block,l/2,l/2-w_1,-d,0,w/2,w/2+l_1  !torsion length 1
                block,l/2,l/2-(w_1+l_2+w_1),-d,0,w/2+l_1,w/2+l_1+w_2
!bending length
                block,l/2-(w_1+l_2+w_1),l/2-(w_1+l_2+w_1)+w_1,-
d,0,w/2+l_1+w_2,w/2+l_1+w_2+l_1 !torsion length 2
                vglue,all

            !Mesh
                !Discharge Structure/Ribbon
                    vsel,all                      !(select beam volume)
                    aslv                          !(select  areas  in  the
volume)
                    asel,r,loc,y,0
                    lsla                          !(select  lines  in  the
areas)
                    lesize,all,elmsize            !divide  all  lines  into
elements of specified size

                    type,1            !assign element type
                    mat,1       !assign material type
```

```
                        amesh,all   !mesh areas

                        !view the area
                        /eshape,1.0
                        /EDGE,1,0,45
                        /GLINE,1,0
                        /replot

                  !Air Gap
                        msha,1,3d                ! set element shape to tet
                        mshmid,2                 ! drop mid-side nodes

                        vsel,all
                        aslv
                        lsla
                        lsel,r,loc,y,-d
                        lesize,all,elmsize         !divide  bottom  lines  into
elements of specified size
                        lsla
                        lsel,r,loc,y,-d/2
                        lesize,all,,,3          !divide   thickness   into   1
element

                        type,2
                        mat,2
                        vmesh,5,8

                        !view the area
                        /view,1,1,1,1
                        /number,1
                        /pnum,type,1
                        eplot

FINISH
/SOLU

      ANTYPE,TRANS
      TRNOPT,FULL

      ! First Load Step
            !Boundary Conditions
                  !Make symmetric about origin X-Z
                        asel,s,loc,z,0
                        asel,a,loc,x,0
                        aplo
                        DA,all,SYMM

                  !structural BC on connected area
                        asel,s,loc,z,w_tot
                        aplot
                        DA, all, UX , 0
                        DA, all, UY , 0
                        DA, all, UZ , 0

                  !structural BC on bottom areas
                        asel,s,loc,y,-d
```

```
                  aplot
                  DA, all, UX , 0
                  DA, all, UY , 0
                  DA, all, UZ , 0

            ! electrical BC on bottom areas
                  asel,s,loc,y,-d
                  aplot
                  da,all,volt,0           ! ground

            ! electrical BC on top areas
                  asel,s,loc,y,0
                  aplot
                  da,all,volt,V           ! electrode

            asel,all
            aplo

      !Output data
            outres,all,all

      !Nonlinear geometry
            nlgeom,on

      !Convergence
            cnvtol,f,1,1e-6         !set tolerance on force convergence
            neqit,50                !set   max   number   of   equilibrium
iterations

      !Timing
            time,10e-8
            autots,ON
            deltim,time_int!,time_int/10,time_int*100,ON
            kbc,1               ! stepped loading

      !Write 1st load step
            lswrite

! Second Load Step
      !Boundary Conditions
            !Make symmetric about origin X-Z
                  asel,s,loc,z,0
                  asel,a,loc,x,0
                  aplo
                  DA,all,SYMM

            !structural BC on connected area
                  asel,s,loc,z,w_tot
                  aplot
                  DA, all, UX , 0
                  DA, all, UY , 0
                  DA, all, UZ , 0

            !structural BC on bottom areas
                  asel,s,loc,y,-d
                  aplot
```

```
                  DA, all, UX , 0
                  DA, all, UY , 0
                  DA, all, UZ , 0

            ! electrical BC on bottom areas
                  asel,s,loc,y,-d
                  aplot
                  da,all,volt,0           ! ground

            ! electrical BC on top areas
                  asel,s,loc,y,0
                  aplot
                  da,all,volt,0           ! electrode

      !Output data
            outres,all,all

      !Nonlinear geometry
            nlgeom,on

      !Convergence
            cnvtol,f,1,1e-6         !set tolerance on force convergence
            neqit,50                !set   max   number   of   equilibrium
iterations

      !Timing
            time,11e-8
            autots,ON
            deltim,time_int,time_int/10,time_int*100,ON   !comment   out
after second term for better results
            kbc,1               ! stepped loading

      !Write 2nd load step---------------------------------
            lswrite

      save

!Solve
      lssolve,1,2
```

Appendix C

LAMMPS Scripts

This appendix contains the scripts used to run the LAMMPS simulations referred to in Chapter 4. All scripts will run with LAMMPS release 16Mar2018. OVITO can be used for post-processing with the dump files, as well as MATLAB. Note the use of word wrap for lines exceeding the page width in this document. The scripts below are used for simulating the 2B ribbon design.

C1. Ribbon Geometry (geom_2B.txt)

Sections of this script are read into LAMMPS by the main LAMMPS input script to generate the atom positions and atom grouping for loads and post-processing. This script is referenced as 'geom_2B.txt' by the main script.

```
label vars

# Graphene Geometry Variables
variable latScale_graphene equal 2.46 #Angstroms/Graphene Lattice

variable graphene_A_width equal ${Aw}
variable graphene_A_length equal ${Al}
variable graphene_A_xlo equal "-v_graphene_A_width / 2"
variable graphene_A_xhi equal "v_graphene_A_width / 2"
variable graphene_A_ylo equal "-v_graphene_A_length / 2"
variable graphene_A_yhi equal "v_graphene_A_length / 2"

variable graphene_B_width equal ${Bw}
variable graphene_B_length equal ${Bl}
variable graphene_B_xlo equal "v_graphene_A_xhi - v_graphene_B_width"
variable graphene_B_xhi equal "v_graphene_A_xhi"
variable graphene_B_yhi equal "v_graphene_A_yhi + v_graphene_B_length"
variable graphene_B_ylo equal "v_graphene_A_yhi"

variable graphene_C_width equal ${Cw}
variable graphene_C_length equal ${Cl}
variable graphene_C_xlo equal "v_graphene_B_xhi - v_graphene_C_width"
variable graphene_C_xhi equal "v_graphene_B_xhi"
```

```
variable graphene_C_yhi equal "v_graphene_B_yhi + v_graphene_C_length"
variable graphene_C_ylo equal "v_graphene_B_yhi"

variable graphene_D_width equal ${Bw}
variable graphene_D_length equal ${Bl}
variable graphene_D_xlo equal "v_graphene_C_xlo"
variable graphene_D_xhi equal "v_graphene_C_xlo + v_graphene_D_width"
variable graphene_D_yhi equal "v_graphene_C_yhi + v_graphene_D_length"
variable graphene_D_ylo equal "v_graphene_C_yhi"

variable graphene_E_width equal ${Cw}
variable graphene_E_length equal ${Cl}
variable graphene_E_xlo equal "v_graphene_D_xlo"
variable graphene_E_xhi equal "v_graphene_D_xlo + v_graphene_E_width"
variable graphene_E_yhi equal "v_graphene_D_yhi + v_graphene_E_length"
variable graphene_E_ylo equal "v_graphene_D_yhi"

variable graphene_F_width equal ${Bw}
variable graphene_F_length equal ${Bl}
variable graphene_F_xlo equal "v_graphene_E_xhi - v_graphene_F_width"
variable graphene_F_xhi equal "v_graphene_E_xhi"
variable graphene_F_yhi equal "v_graphene_E_yhi + v_graphene_F_length"
variable graphene_F_ylo equal "v_graphene_E_yhi"

variable graphene_mobile_ylo equal "-v_graphene_F_yhi"
variable graphene_mobile_yhi equal "v_graphene_F_yhi"
variable graphene_xlo equal "-v_graphene_F_xhi"
variable graphene_xhi equal "v_graphene_F_xhi"
variable graphene_ylo equal "v_graphene_mobile_ylo - 5"
variable graphene_yhi equal "v_graphene_mobile_yhi + 5"
variable graphene_zlo equal "-1"
variable graphene_zhi equal "1"

variable     graphene_mobile_length     equal     "v_graphene_mobile_yhi     -
v_graphene_mobile_ylo" #Angstroms
variable graphene_length equal "v_graphene_mobile_length + 10" #Angstroms
variable graphene_width equal "v_graphene_xhi - v_graphene_xlo" #Angstroms


# Simulation Box Geometry
variable sim_xlo equal "v_graphene_xlo*1.1"
variable sim_xhi equal "v_graphene_xhi*1.1"
variable sim_ylo equal "v_graphene_ylo*1.1"
variable sim_yhi equal "v_graphene_yhi*1.1"
variable sim_zlo equal "v_graphene_zlo*2"
variable sim_zhi equal "v_graphene_zhi*2"

jump in.eq.2B end_geom_vars
################################################
label lat_reg_gro

# GEOMETRY --------------------------------------
# Graphene Regions and lattice
lattice custom ${latScale_graphene} &
            a1 1 0 0 &
            a2 0 1.732050807569 0 &
```

```
            a3 0 0 3.35 &
            basis 0 0 0 &
            basis 0.5 0.166666666667 0 &
            basis 0.5 0.5 0 &
            basis 0.0 0.666666666667 0

region      graphene_region_A block &
            ${graphene_A_xlo} ${graphene_A_xhi} &
            ${graphene_A_ylo} ${graphene_A_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_region_B1 block &
            ${graphene_B_xlo} ${graphene_B_xhi} &
            ${graphene_B_ylo} ${graphene_B_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_C1 block &
            ${graphene_C_xlo} ${graphene_C_xhi} &
            ${graphene_C_ylo} ${graphene_C_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_D1 block &
            ${graphene_D_xlo} ${graphene_D_xhi} &
            ${graphene_D_ylo} ${graphene_D_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_E1 block &
            ${graphene_E_xlo} ${graphene_E_xhi} &
            ${graphene_E_ylo} ${graphene_E_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_F1 block &
            ${graphene_F_xlo} ${graphene_F_xhi} &
            ${graphene_F_ylo} ${graphene_F_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_region_B2 block &
            ${graphene_B_xlo} ${graphene_B_xhi} &
            -${graphene_B_yhi} -${graphene_B_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_C2 block &
            ${graphene_C_xlo} ${graphene_C_xhi} &
            -${graphene_C_yhi} -${graphene_C_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_D2 block &
            ${graphene_D_xlo} ${graphene_D_xhi} &
            -${graphene_D_yhi} -${graphene_D_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_E2 block &
            ${graphene_E_xlo} ${graphene_E_xhi} &
            -${graphene_E_yhi} -${graphene_E_ylo} &
```

```
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_F2 block &
            ${graphene_F_xlo} ${graphene_F_xhi} &
            -${graphene_F_yhi} -${graphene_F_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_region_B3 block &
            -${graphene_B_xhi} -${graphene_B_xlo} &
            -${graphene_B_yhi} -${graphene_B_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_C3 block &
            -${graphene_C_xhi} -${graphene_C_xlo} &
            -${graphene_C_yhi} -${graphene_C_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_D3 block &
            -${graphene_D_xhi} -${graphene_D_xlo} &
            -${graphene_D_yhi} -${graphene_D_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_E3 block &
            -${graphene_E_xhi} -${graphene_E_xlo} &
            -${graphene_E_yhi} -${graphene_E_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_F3 block &
            -${graphene_F_xhi} -${graphene_F_xlo} &
            -${graphene_F_yhi} -${graphene_F_ylo} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_region_B4 block &
            -${graphene_B_xhi} -${graphene_B_xlo} &
            ${graphene_B_ylo} ${graphene_B_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_C4 block &
            -${graphene_C_xhi} -${graphene_C_xlo} &
            ${graphene_C_ylo} ${graphene_C_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_D4 block &
            -${graphene_D_xhi} -${graphene_D_xlo} &
            ${graphene_D_ylo} ${graphene_D_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_E4 block &
            -${graphene_E_xhi} -${graphene_E_xlo} &
            ${graphene_E_ylo} ${graphene_E_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_F4 block &
            -${graphene_F_xhi} -${graphene_F_xlo} &
```

```
            ${graphene_F_ylo} ${graphene_F_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_region_fixed1 block &
            ${graphene_F_xlo} ${graphene_F_xhi} &
            ${graphene_F_yhi} ${graphene_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_fixed2 block &
            ${graphene_F_xlo} ${graphene_F_xhi} &
            -${graphene_yhi} -${graphene_F_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_fixed3 block &
            -${graphene_F_xhi} -${graphene_F_xlo} &
            ${graphene_F_yhi} ${graphene_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box
region      graphene_region_fixed4 block &
            -${graphene_F_xhi} -${graphene_F_xlo} &
            -${graphene_yhi} -${graphene_F_yhi} &
            ${graphene_zlo} ${graphene_zhi} &
            units box

region      graphene_arm1   union   5   graphene_region_B1   graphene_region_C1
graphene_region_D1 graphene_region_E1 graphene_region_F1
region      graphene_arm2   union   5   graphene_region_B2   graphene_region_C2
graphene_region_D2 graphene_region_E2 graphene_region_F2
region      graphene_arm3   union   5   graphene_region_B3   graphene_region_C3
graphene_region_D3 graphene_region_E3 graphene_region_F3
region      graphene_arm4   union   5   graphene_region_B4   graphene_region_C4
graphene_region_D4 graphene_region_E4 graphene_region_F4
region      graphene_fixed        union        4        graphene_region_fixed1
graphene_region_fixed2 graphene_region_fixed3 graphene_region_fixed4
region      graphene_region    union    6    graphene_region_A    graphene_arm1
graphene_arm2 graphene_arm3 graphene_arm4 graphene_fixed

create_atoms      1 region graphene_region

# GROUPS ----------------------------------------

group graphene_group type 1
region      graphene_mobile_bounds block &
            INF   INF   &
            ${graphene_mobile_ylo}  ${graphene_mobile_yhi}  &
            INF   INF   &
            units box
group graphene_mobile_bounds region graphene_mobile_bounds
group graphene_mobile intersect graphene_group graphene_mobile_bounds
group       graphene_fixed region graphene_fixed
group graphene_stage region graphene_region_A

jump in.eq.2B end_geom_lat_reg_gro
```

C2. Dynamic Equilibrium Input (in.eq.2B)

This is an input script for running a dynamic equilibrium simulation of a 2B ribbon in LAMMPS. When jumping to the geometry file, it is referenced as 'in.eq.2B'.

```
### Simulation of a graphene ribbon being pulled into an Si
### substrate by electrostatic charges
### INPUTS: voltage, ribbon geometry parameters, trench depth, ribbon geometry
generator files, AIREBO potential file
### OUTPUTS: dump.*.txt, parameters.txt, energy.txt

### INITIAL SETUP ################################
      # VARIABLES -------------------------------------
            variable inputScriptName string in.eq.2B #name of this file so it
can be looped back into during parameter sweeps

            # Parameters
                  variable Cl equal 30 #Ribbon Length Value to apply    "b"
                  variable Cw equal 90 #Ribbon Width Value to apply     "L"
                  variable Bl equal 30 #Ribbon Length Value to apply    "e"
                  variable Bw equal 30 #Ribbon Width Value to apply     "t"
                  variable Al equal 90 #Ribbon Length Value to apply    "w"
                  variable Aw equal 210 #Ribbon Width Value to apply    "l"
                  variable design string 2B #Old 2B 1B TS  #Ribbon Design

            # TIMING
                  variable timestep equal 1e-3 #0.0025 #Picoseconds
                  variable time_relax equal 10000e3 #Timesteps

            # Graphene Geometry
                  jump geom_${design}.txt vars #jump to corresponding graphene
geometry file
                  label end_geom_vars

            # Post-Processing
                  log
log.${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}.txt
                  variable                outputFolder                string
${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}
                  variable dumpRate_energy string 1000

      # SIMULATION SETTINGS -----------------------------
            units metal
            dimension 3
            atom_style atomic

            neigh_modify delay 0 every 1 check yes

            # Simulation Box
                  boundary f f m
                  region      sim_region block &
                              ${sim_xlo} ${sim_xhi} &
                              ${sim_ylo} ${sim_yhi} &
```

```
                              ${sim_zlo} ${sim_zhi} &
                              units box
                  create_box  1 sim_region

      # ATOM PROPERTIES --------------------------------
            # Masses
            mass  1 12.011 #C

      # GEOMETRY ---------------------------------------
            jump geom_${design}.txt lat_reg_gro #jump to corresponding graphene
geometry file
            label end_geom_lat_reg_gro

            #exclude the calculation of pairwise interactions within the fixed
group
            neigh_modify exclude group graphene_fixed graphene_fixed

      # POST-PROCESSING --------------------------------
            shell rd ${outputFolder} /s /q
            shell mkdir ${outputFolder}
            shell cd ${outputFolder}

            # variable to track atom position
            variable graphene_pos_min equal bound(graphene_mobile,zmin)
            compute     graphene_pos_ave graphene_mobile reduce ave z
            compute     graphene_pos_max graphene_mobile reduce max z
            compute     graphene_pos_stage_min graphene_stage reduce min z
            compute     graphene_pos_stage_ave graphene_stage reduce ave z
            compute     graphene_pos_stage_max graphene_stage reduce max z

            #set up to output any parameters from the whole simulation
            print "SIMULATION                  PARAMETERS"                 file
parameters${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}.txt
            #           "Parameter,Value,Units" append filename.txt
            print "numAtoms,$(count(graphene_mobile)),Atoms"            append
parameters${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}.txt

            shell cd ..

### RUN - RELAX ###################################
      # VARIABLES --------------------------------------

      # POTENTIALS -------------------------------------
            pair_style airebo 3.0
            pair_coeff * * CH.airebo.1_92 C

      # FIXES & CONDITIONS -----------------------------
            #give an initial velocity to graphene atoms and maintain it using
the nvt ensemble
            velocity    graphene_mobile create 300 12345
            fix         1 all nvt temp 300 300 0.05

            #fix the ends of the graphene ribbon
            fix   graphene_fix_fixed graphene_fixed setforce 0.0 0.0 0.0

      # DUMPS ------------------------------------------
```

```
            #output per-atom position and other values
            #dump ovito all custom 100 dump.*.txt id element x y z
            #dump_modify ovito element C

            shell cd ${outputFolder}
            #output global energies
            fix   print_energy all print ${dumpRate_energy} "$(time) $(step)
$(temp)  $(pe)  $(ke)  $(evdwl)  $(v_graphene_pos_min)  $(c_graphene_pos_ave)
$(c_graphene_pos_max) $(c_graphene_pos_stage_min) $(c_graphene_pos_stage_ave)
$(c_graphene_pos_stage_max) $(count(graphene_mobile))" &
                        append
energyRELAX${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}.txt screen no title "time
step temp pe ke evdwl graphene_pos_min graphene_pos_ave graphene_pos_max
c_graphene_pos_stage_min   c_graphene_pos_stage_ave   c_graphene_pos_stage_max
numAtoms"

            restart     200000 2Bbeq.*.restart

      # RUN ---------------------------------------------
            timestep    ${timestep}
            thermo            1000
            thermo_style      custom    time    step    temp    pe    ke    evdwl
v_graphene_pos_min c_graphene_pos_stage_ave vol
            thermo_modify     lost warn
            run   ${time_relax}

      # CLEANUP -----------------------------------------
            shell cd ..
            unfix       print_energy
            undump      ovito

### POST-PROCESSING ##############################

      shell cd ${outputFolder}

      #output any parameters from the whole simulation
      #           "Parameter,Value,Units" append filename.txt
      print "numAtomsFinal,$(count(graphene_mobile)),Atoms"             append
parameters${design}_eqBl${Bl}Bw${Bw}Al${Al}Aw${Aw}.txt

      shell cd ..
```

C3. Pull-in Voltage Determination Input (in.Vp.2B)

This script loops as it decreases the voltage at the user-specified increment. Once the ribbon response oscillates instead of experiencing pull-in, the simulation will run until the time limit expires. It references itself as ‘in.Vp.2B’.

```
### Simulation of a graphene
```

```
### INPUTS: voltage, ribbon geometry parameters, trench depth, dynamic
equilibrium restart file, AIREBO potential file
### OUTPUTS: dump.*.txt, parameters.txt, log file

### INITIAL SETUP ################################
      # VARIABLES -----------------------------------
            variable inputScriptName string in.Vp.2B #name of this file so it
can be looped back into during parameter sweeps

            # Inputs
                  variable Cl equal 30 #Ribbon Length Values to apply   "b"
                  variable Cw equal 90    #Ribbon Width Values to apply "L"
                  variable Bl equal 30  #Ribbon Length Values to apply  "e"
                  variable Bw equal 30 #Ribbon Width Values to apply    "t"
                  variable Al equal 90   #Ribbon Length Values to apply "w"
                  variable Aw equal 210   #Ribbon Width Values to apply "l"
                  variable d equal 150 #Trench Depth Values to apply    "d"
                  variable design string 2B #Old 2B 1B TS  #Ribbon Design

            # Electrical Parameters
                  variable epsilon equal "1 * 8.854e-12 * 6.242e18 / 1e10"
#protoncharge/V/Angs (F/m => (C/V)/m * 6.242e18protons/1C * 1m/1e10Angs)

            # Trench Geometry
                  variable trench_z equal $d #Angstroms
                  variable pullin_z equal "1 - v_d"
                  variable upper_z  equal 60

            # Timing
                  variable timestep equal 0.001 #Picoseconds
                  variable time_pullin equal 1e6 #Timesteps

            # Initialize Voltage Loop Variables
                  variable newV equal 100
                  variable Vstep equal 10
                  variable iter equal 0

###  LOOP - FIND V-PULLIN ################################

      label loop_Vp
      clear

      # Voltage and Iteration Assignment
            variable voltage equal ${newV}

            variable iter_last equal ${iter}
            variable iter equal "v_iter_last+1"

      # Post Processing
            log
log.${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d.txt
            variable                    outputFolder                    string
${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d
            print "new voltage: ${newV}"

      # Simuation Settings
```

```
            read_restart 2Bbeq.*.restart

            neigh_modify delay 0 every 1 check yes
            neigh_modify exclude group graphene_fixed graphene_fixed

            # Simulation Box
                  change_box  all &
                                    z final -$d ${upper_z} &
                                    boundary f f f

      # Computes
            # variables to track atom position
            variable    numAtoms equal "count(graphene_mobile)"
            variable    graphene_pos_min equal bound(graphene_mobile,zmin)
            compute     graphene_pos_ave graphene_mobile reduce ave z
            compute     graphene_pos_stage_min graphene_stage reduce min z
            compute     graphene_pos_stage_ave graphene_stage reduce ave z

      # Post Processing
            dump  ovito all custom 100 dump.*.txt id element x y z
            dump_modify ovito element C

            #set up to output any parameters from the whole simulation
            print "SIMULATION                  PARAMETERS"                  file
parameters${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d.txt
            #           "Parameter,Value,Units" append filename.txt
            print "iteration,${iter},OfVoltageFinding"                  append
parameters${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d.txt
            print "voltage,${voltage},Volts"                            append
parameters${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d.txt
            print "numAtoms,${numAtoms},Atoms"                          append
parameters${design}_i${iter}Bl${Bl}Bw${Bw}Al${Al}Aw${Aw}d$d.txt

      # Fixes
            fix         1 all nvt temp 300 300 0.05

            fix   graphene_fix_fixed graphene_fixed setforce 0.0 0.0 0.0

            variable graphene_atom_A equal 2.47678#01857585101111 #Angstroms^2
(based on area of a graphene sheet divided by no. atoms in that sheet)
            #variable  q_graphene  atom  "v_epsilon  *  v_graphene_atom_A  *
v_voltage / z"
            #variable E_graphene atom "v_voltage / (2*z)"
            variable   F_elec   atom   "-(v_epsilon   *   v_graphene_atom_A   *
(v_voltage)^2) / (2 * (v_d+z)^2)"
            variable   U_elec   atom   "-(v_epsilon   *   v_graphene_atom_A   *
(v_voltage)^2) / (2 * (v_d+z))"
            fix ElecField graphene_mobile addforce 0 0 v_F_elec energy v_U_elec
            fix_modify ElecField energy yes #include the potential energy from
the Efield in calculations

            fix halt_pullin all halt 1 v_graphene_pos_min < ${pullin_z}

      # POTENTIALS ---------------------------------
            pair_style airebo 3.0
            pair_coeff * * CH.airebo.1_92 C
```

```
      # Run ---------------------------------
            shell rd ${outputFolder} /s /q
            shell mkdir ${outputFolder}
            shell cd ${outputFolder}

            thermo            10
            thermo_style      custom time step temp pe ke evdwl f_ElecField
v_graphene_pos_min         c_graphene_pos_ave         c_graphene_pos_stage_min
c_graphene_pos_stage_ave v_numAtoms
            thermo_modify     lost warn
            run ${time_pullin}

            shell cd ..

      # Loop Back
            variable newV equal "v_voltage - v_Vstep"
            undump ovito
            jump ${inputScriptName} loop_Vp
```

Appendix D

MDOF MATLAB Scripts

This appendix includes the MATLAB functions used with the MDOF model from Chapter 2 to run a dynamic simulation of pull-in for each ribbon design. The functions 'PatchPlotConstraint.m' and 'EulerStiffnessMatrix.m' called in 'Vpullin_calculator_FACT.m' are available from Lucas Shaw and Jonathan Hopkins, respectively.

D1. FACT Pull-in Simulator (Vpullin_calculator_FACT.m)

This script calculates the pull-in voltage, response time at that voltage and natural frequencies and mode shapes for the user-specified design with the user-specified dimensions. It contains sections that create input matrices for 'EulerStiffnessMatrix.m' and 'MassMatrix.m'. MATLAB's ode45 function is run iteratively to find the pull-in voltage using the 'dynamic_pullin_FACT.m' function.

```
clc
clear all
close all

%units are according to the umksV system
%% Parameters
%Material Properties
E = 680e3; %MPa [(kg)/(um)(s2)] Young's Modulus
G = 280e3; %MPa Shear Modulus
h = 3.4e-4; %um Ribbon Thickness (governed by 2D material thickness)
rho_2D = .77e-24; %kg/um^2 2D mass density of material
rho = rho_2D/h; %3D mass density
C_nonP = 4900; %nonpristine graphene correction factor

%Electrical Conditions
V_hi = 10; %V (upper bound guess for pull-in voltage)
V_lo = 0; %V (lower bound guess for pull-in voltage)
epsilon_0 = 8.854e-6; %pF/um Vacuum permittivity

%Timing Conditions
tlim = 1e-8; %seconds
```

```
%% Ribbon Geometry Parameters
% indicate which ribbon design is to be studied (Old/TS/1B/2B)
design = 'TS';

switch design
    case 'Old'
        %as fabricated dimensions
        L = 7; %um
        b = 5; %um
        d = .35; %um
        h = 3.4e-4; %um
        E = 600e3; %MPa
        G = 280e3;
        Nstages = 1;
        %FACT stage and flexure dimensions
        L_s = L/1000;
        L_f = L/2 - L_s(1)/2;

        %effective FACT dimensions to compensate for flexure mass and area
        %only comes into effect in generating the mass matrix and the area
        %matrix (which determine inertial forces and electrostatic forces)
        L_seff = L/2;  %include half of the flexure area on each side (this
matches up with the beam theory results for fixed-fixed)
        A_seff = L_seff(1)*b;

    case 'TS'
        %as fabricated dimensions
        w = 3; %um
        l = 15; %um
        t = 3; %um
        e = 3; %um
        d = 1; %um

        Nstages = 3;
        %FACT stage and flexure dimensions
        L_s = [l/1000; t; t];
        b_s = [w; w; w];
        L_f = [(l/2 - L_s(1)/2 - t); (l/2 - L_s(1)/2 - t); e; e; e; e];
        b_f = [w; w; t; t; t; t];
        %effective FACT dimensions
        A_seff = [(L_s(1)*b_s(1) + L_f(1)*b_f(1)/2 + L_f(2)*b_f(2)/2);
                    2*L_s(2)*b_s(2);
                    2*L_s(3)*b_s(3);
                    ];
        L_seff     =     [A_seff(1)/b_s(1);     sqrt(A_seff(2)*L_s(2)/b_s(2));
sqrt(A_seff(3)*L_s(3)/b_s(3))];
        b_seff = [b_s(1); L_seff(2)*b_s(2)/L_s(2); L_seff(3)*b_s(3)/L_s(3)];

    case '1B'
        %as fabricated dimensions
        l = 15; %um
        w = 9; %um
        L = 3; %um
        b = 3; %um
        t = 3; %um
```

```
        e = 3; %um
        d = 1; %um
        Nstages = 11;
        %FACT stage and flexure dimensions
        L_s = [l/1000; t; t; e; e; e; e; e; e; e; e]; %vector of the Lengths of
each stage
        b_s = [w; w; w; b; b; b; b; b; b; b; b]; %vector of the widths of each
stage
        L_f = [l/2-L_s(1)/2; l/2-L_s(1)/2; e; e; e; e; L; L; L; L; e; e; e; e];
%vector of the Lengths of each flexure
        b_f = [w; w; t; t; t; t; b; b; b; b; t; t; t; t]; %vector of the widths
of each flexure

        %effective FACT dimensions (for Dynamics EOM-- ES pull-in area and mass
inertia)
        A_seff = L_s .* b_s *2;
            A_seff(1) = (L_s(1)*b_s(1) + L_f(1)*b_f(1)/2 + L_f(2)*b_f(2)/2);
        L_seff = L_s;   %sqrt(A_seff(2)*L_s(2)/b_s(2));
            L_seff(1) = A_seff(1)/b_s(1);
        b_seff = b_s;   %L_seff(2)*b_s(2)/L_s(2);
    case '2B'
        %as fabricated dimensions
        l = 15; %um
        w = 9; %um
        L = 3; %um
        b = 3; %um
        t = 3; %um
        e = 3; %um
        d = 1; %um
        Nstages = 19;
        %FACT stage and flexure dimensions
        L_s = [l/1000; t; t; e; e; e; e; e; e; e; e; e; e; e; e; e; e; e; e];
%vector of the Lengths of each stage
        b_s = [w; w; w; b; b; b; b; b; b; b; b; b; b; b; b; b; b; b; b]; %vector
of the widths of each stage
        L_f = [l/2-L_s(1)/2; l/2-L_s(1)/2; e; e; e; e; L; L; L; L; e; e; e; e;
L; L; L; L; e; e; e; e]; %vector of the Lengths of each flexure
        b_f = [w; w; t; t; t; t; b; b; b; b; t; t; t; t; b; b; b; b; t; t; t;
t]; %vector of the widths of each flexure

        %effective FACT dimensions (for ES pull-in area and mass inertia)
        A_seff = L_s .* b_s *2;
            A_seff(1) = (L_s(1)*b_s(1) + L_f(1)*b_f(1)/2 + L_f(2)*b_f(2)/2);
        L_seff = L_s;   %sqrt(A_seff(2)*L_s(2)/b_s(2));
            L_seff(1) = A_seff(1)/b_s(1);
        b_seff = b_s;   %L_seff(2)*b_s(2)/L_s(2);
end

%% Generate Euler Stiffnes Matrix and Stage Mass Matrix
switch design
    case 'Old'
        Constraint = [
            [L_s(1)/2 0 0] [-1 0 0]   [0 0 1]     L_f(1) b h E G 1 0
            [-L_s(1)/2 0 0]  [1 0 0]    [0 0 1]     L_f(1) b h E G 1 0
            1   0   2    0 0 0      0 0 0       0    0 0 0 0 0 0
                    ];
```

```
        StageParams =   [
                        [0 0 0] [0 0 1] [0 1 0] L_seff(1) b h rho
                        ];
    case 'TS'
        Constraint = [
            [0 -L_s(1)/2 0]   [0 1 0]   [0 0 1]     L_f(1) b_f(1) h E G 1 2
            [0 L_s(1)/2 0]  [0 -1 0]    [0 0 1]     L_f(2) b_f(2) h E G 1 3
            [b_s(2)/2 -(L_s(1)/2+L_f(1)+L_s(2)/2) 0]   [-1 0 0]      [0 0 1]
L_f(3) b_f(3) h E G 2 0
            [-b_s(2)/2 -(L_s(1)/2+L_f(1)+L_s(2)/2) 0]   [1 0 0]      [0 0 1]
L_f(4) b_f(4) h E G 2 0
            [-b_s(3)/2 (L_s(1)/2+L_f(2)+L_s(3)/2) 0]   [1 0 0]      [0 0 1]
L_f(5) b_f(5) h E G 3 0
            [b_s(3)/2 (L_s(1)/2+L_f(2)+L_s(3)/2) 0]   [-1 0 0]      [0 0 1]
L_f(6) b_f(6) h E G 3 0
            3   0   6    0 0 0      0 0 0       0    0 0 0 0 0 0
                      ];
        StageParams =   [
                        [0 0 0] [0 0 1] [1 0 0] L_seff(1) b_seff(1) h rho
                        [0 -(L_s(1)/2+L_f(1)+L_s(2)/2) 0] [0 0 1] [1 0 0]
L_seff(2) b_seff(2) h rho
                        [0 (L_s(1)/2+L_f(2)+L_s(3)/2) 0] [0 0 1] [1 0 0]
L_seff(3) b_seff(3) h rho
                        ];
    case '1B'
        Constraint = [
            [0 -L_s(1)/2 0]   [0 1 0]   [0 0 1]     L_f(1) b_f(1) h E G 1 2
            [0 L_s(1)/2 0]  [0 -1 0]    [0 0 1]     L_f(2) b_f(2) h E G 1 3
            [b_s(2)/2 -(L_s(1)/2+L_f(1)+L_s(2)/2) 0]   [-1 0 0]      [0 0 1]
L_f(3) b_f(3) h E G 2 4
            [-b_s(2)/2 -(L_s(1)/2+L_f(1)+L_s(2)/2) 0]   [1 0 0]      [0 0 1]
L_f(4) b_f(4) h E G 2 5
            [-b_s(3)/2 (L_s(1)/2+L_f(2)+L_s(3)/2) 0]   [1 0 0]      [0 0 1]
L_f(5) b_f(5) h E G 3 6
            [b_s(3)/2 (L_s(1)/2+L_f(2)+L_s(3)/2) 0]   [-1 0 0]      [0 0 1]
L_f(6) b_f(6) h E G 3 7
            [b_s(2)/2+L_f(3)+b_f(7)/2 -(L_s(1)/2+L_f(1)) 0] [0 -1 0] [0 0 1]
L_f(7) b_f(7) h E G 4 8
            [-(b_s(2)/2+L_f(3)+b_f(7)/2) -(L_s(1)/2+L_f(1)) 0] [0 -1 0] [0 0 1]
L_f(8) b_f(8) h E G 5 9
            [-(b_s(2)/2+L_f(3)+b_f(7)/2) (L_s(1)/2+L_f(1)) 0] [0 1 0] [0 0 1]
L_f(9) b_f(9) h E G 6 10
            [b_s(2)/2+L_f(3)+b_f(7)/2 (L_s(1)/2+L_f(1)) 0] [0 1 0] [0 0 1]
L_f(10) b_f(10) h E G 7 11
            [b_s(2)/2+L_f(3)+b_f(7) -(L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2) 0]  [-1
0 0]    [0 0 1]     L_f(11) b_f(11) h E G 8 0
            [-(b_s(2)/2+L_f(3)+b_f(7))  -(L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2)  0]
[1 0 0]    [0 0 1]     L_f(12) b_f(12) h E G 9 0
            [-(b_s(2)/2+L_f(3)+b_f(7))  (L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2)  0]
[1 0 0]    [0 0 1]     L_f(13) b_f(13) h E G 10 0
            [b_s(2)/2+L_f(3)+b_f(7) (L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2) 0]  [-1
0 0]    [0 0 1]     L_f(14) b_f(14) h E G 11 0
            11   0   14    0 0 0      0 0 0       0    0 0 0 0 0 0
                      ];
        StageParams =   [
                        [0 0 0] [0 0 1] [1 0 0] L_seff(1) b_seff(1) h rho
```

```
                        [0  -(L_s(1)/2+L_f(1)+L_s(2)/2)  0]  [0  0  1]  [1  0  0]
L_seff(2) b_seff(2) h rho
                        [0  (L_s(1)/2+L_f(2)+L_s(3)/2)  0]  [0  0  1]  [1  0  0]
L_seff(3) b_seff(3) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)   (L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(4) b_seff(4) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)  (L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(5) b_seff(5) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2) -(L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(6) b_seff(6) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)  -(L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(7) b_seff(7) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)     (L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)) 0] [0 0 1] [1 0 0] L_seff(8) b_seff(8) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)    (L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)) 0] [0 0 1] [1 0 0] L_seff(9) b_seff(9) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)   -(L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)) 0] [0 0 1] [1 0 0] L_seff(10) b_seff(10) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)    -(L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)) 0] [0 0 1] [1 0 0] L_seff(11) b_seff(11) h rho
                        ];
    case '2B'
        Constraint = [
            [0 -L_s(1)/2 0]   [0 1 0]   [0 0 1]     L_f(1) b_f(1) h E G 1 2
            [0 L_s(1)/2 0]  [0 -1 0]    [0 0 1]     L_f(2) b_f(2) h E G 1 3
            [b_s(2)/2  -(L_s(1)/2+L_f(1)+L_s(2)/2)  0]   [-1  0  0]      [0  0  1]
L_f(3) b_f(3) h E G 2 4
            [-b_s(2)/2  -(L_s(1)/2+L_f(1)+L_s(2)/2)  0]   [1  0  0]      [0  0  1]
L_f(4) b_f(4) h E G 2 5
            [-b_s(3)/2  (L_s(1)/2+L_f(2)+L_s(3)/2)  0]   [1  0  0]      [0  0  1]
L_f(5) b_f(5) h E G 3 6
            [b_s(3)/2  (L_s(1)/2+L_f(2)+L_s(3)/2)  0]   [-1  0  0]      [0  0  1]
L_f(6) b_f(6) h E G 3 7
            [b_s(2)/2+L_f(3)+b_f(7)/2 -(L_s(1)/2+L_f(1)) 0] [0 -1 0] [0 0 1]
L_f(7) b_f(7) h E G 4 8
            [-(b_s(2)/2+L_f(3)+b_f(7)/2) -(L_s(1)/2+L_f(1)) 0] [0 -1 0] [0 0 1]
L_f(8) b_f(8) h E G 5 9
            [-(b_s(2)/2+L_f(3)+b_f(7)/2) (L_s(1)/2+L_f(1)) 0] [0 1 0] [0 0 1]
L_f(9) b_f(9) h E G 6 10
            [b_s(2)/2+L_f(3)+b_f(7)/2  (L_s(1)/2+L_f(1))  0]  [0  1  0]  [0  0  1]
L_f(10) b_f(10) h E G 7 11
            [b_s(2)/2+L_s(3)+b_f(7) -(L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2) 0]  [-1
0 0]    [0 0 1]     L_f(11) b_f(11) h E G 8 12
            [-(b_s(2)/2+L_s(3)+b_f(7))  -(L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2)  0]
[1 0 0]    [0 0 1]     L_f(12) b_f(12) h E G 9 13
            [-(b_s(2)/2+L_s(3)+b_f(7))   (L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2)  0]
[1 0 0]    [0 0 1]     L_f(13) b_f(13) h E G 10 14
            [b_s(2)/2+L_s(3)+b_f(7) (L_s(1)/2+L_f(1)-L_f(7)-L_s(8)/2) 0]  [-1
0 0]    [0 0 1]     L_f(14) b_f(14) h E G 11 15
            [b_s(2)/2+L_f(3)+b_f(7)+L_f(11)+b_f(15)/2       -(L_s(1)/2+L_f(1)-
L_f(7)) 0] [0 1 0] [0 0 1] L_f(15) b_f(15) h E G 12 16
            [-(b_s(2)/2+L_f(3)+b_f(7)+L_f(11)+b_f(15)/2)    -(L_s(1)/2+L_f(1)-
L_f(7)) 0] [0 1 0] [0 0 1] L_f(16) b_f(16) h E G 13 17
            [-(b_s(2)/2+L_f(3)+b_f(7)+L_f(11)+b_f(15)/2)     (L_s(1)/2+L_f(1)-
L_f(7)) 0] [0 -1 0] [0 0 1] L_f(17) b_f(17) h E G 14 18
```

```
            [b_s(2)/2+L_f(3)+b_f(7)+L_f(11)+b_f(15)/2 (L_s(1)/2+L_f(1)-L_f(7))
0] [0 -1 0] [0 0 1] L_f(18) b_f(18) h E G 15 19
            [b_s(2)/2+L_s(3)+b_f(7)+L_f(11)+b_f(15)                          -
(L_s(1)/2+L_f(1)+L_s(2)/2) 0]  [-1 0 0]    [0 0 1]     L_f(19) b_f(19) h E G 16
0
            [-(b_s(2)/2+L_s(3)+b_f(7)+L_f(11)+b_f(15))                       -
(L_s(1)/2+L_f(1)+L_s(2)/2) 0]  [1 0 0]    [0 0 1]     L_f(20) b_f(20) h E G 17
0
            [-(b_s(2)/2+L_s(3)+b_f(7)+L_f(11)+b_f(15))
(L_s(1)/2+L_f(1)+L_s(2)/2) 0]  [1 0 0]    [0 0 1]     L_f(21) b_f(21) h E G 18
0
            [b_s(2)/2+L_s(3)+b_f(7)+L_f(11)+b_f(15) (L_s(1)/2+L_f(1)+L_s(2)/2)
0]  [-1 0 0]    [0 0 1]     L_f(22) b_f(22) h E G 19 0
            19   0   22    0 0 0      0 0 0       0    0 0 0 0 0 0
                      ];
        StageParams =   [
                        [0 0 0] [0 0 1] [1 0 0] L_seff(1) b_seff(1) h rho
                        [0  -(L_s(1)/2+L_f(1)+L_s(2)/2)  0]  [0  0  1]  [1  0  0]
L_seff(2) b_seff(2) h rho
                        [0  (L_s(1)/2+L_f(2)+L_s(3)/2)  0]  [0  0  1]  [1  0  0]
L_seff(3) b_seff(3) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)   (L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(4) b_seff(4) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)  (L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(5) b_seff(5) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2) -(L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(6) b_seff(6) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)  -(L_s(1)/2+L_f(2)+L_s(3)/2)
0] [0 0 1] [1 0 0] L_seff(7) b_seff(7) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)     (L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(8) b_seff(8) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)    (L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(9) b_seff(9) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)/2)   -(L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(10) b_seff(10) h rho
                        [(b_s(1)+L_f(1)+b_s(4)/2)    -(L_s(1)/2+L_f(2)-L_f(7)-
L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(11) b_seff(11) h rho
                        [(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(12) b_seff(12) h
rho
                        [-(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(13) b_seff(13) h
rho
                        [-(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)           -
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(14) b_seff(14) h
rho
                        [(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)            -
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)/2) 0] [0 0 1] [1 0 0] L_seff(15) b_seff(15) h
rho
                        [(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)+L_f(15)+L_s(16)/2) 0] [0 0 1] [1 0 0] L_seff(16)
b_seff(16) h rho
                        [-(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)+L_f(15)+L_s(16)/2) 0] [0 0 1] [1 0 0] L_seff(17)
b_seff(17) h rho
```

```
                        [-(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)           -
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)+L_f(15)+L_s(16)/2) 0] [0 0 1] [1 0 0] L_seff(18)
b_seff(18) h rho
                        [(b_s(1)+L_f(1)+b_s(4)+L_f(11)+b_s(12)/2)           -
(L_s(1)/2+L_f(2)-L_f(7)-L_s(8)+L_f(15)+L_s(16)/2) 0] [0 0 1] [1 0 0] L_seff(19)
b_seff(19) h rho
                        ];
end

K = EulerStiffnessMatrix(Constraint)*C_nonP;
M = MassMatrix(StageParams);

%Visualize the flexures in the ribbon geometry
figure(98)
PatchPlotConstraint(Constraint) %function created by Luke Shaw

%% Dynamic System Iterative Solving until finding Vpullin within tolerance
% Parameters and Initial Conditions for the ode solver
tspan = [0 tlim];
T_0 = zeros(6*Nstages,1); %[0; 0; 0; 0; 0; 0];
Tdot_0 = zeros(6*Nstages,1); %[0; 0; 0; 0; 0; 0];
Tddot_0 = zeros(6*Nstages,1); %[0; 0; 0; 0; 0; 0];
x_0 = [T_0; Tdot_0];
go = 1; %flag variable to stop if guesses are unreasonable

%plot all the iterative pull-in responses as the voltage is found
figure(99)
dischargeContactParams = @(t,x) dischargeContact(t,x,d); %define function that
carries the trench depth parameter into the contact event funciton
options = odeset('Events',dischargeContactParams); %use event function to
indicate when stage makes contact with trench bottom

%find response at upper bound voltage guess
[t_resp,x_resp,t_contact]                    =                    ode45(@(t,x)
dynamic_pullin_FACT(t,x,Tdot_0,Tddot_0,M,K,A_seff,d,V_hi,epsilon_0),    tspan,
x_0,options);
z_resp = zeros(size(x_resp,1),Nstages);
for i = 1:Nstages
    z_resp(:,i) = x_resp(:,i*6);
end
semilogx(t_resp,-z_resp,'-b')
xlim([1e-10 tlim])
drawnow
hold on
if max(max(z_resp)) < .99*d
    disp('No Pullin at V_hi, increase V_hi')
    go = 0;
else
%find response at lower bound voltage guess
    [t_resp,x_resp,t_contact]                    =                    ode45(@(t,x)
dynamic_pullin_FACT(t,x,Tdot_0,Tddot_0,M,K,A_seff,d,V_lo,epsilon_0),    tspan,
x_0,options);
    z_resp = zeros(size(x_resp,1),Nstages);
    for i = 1:Nstages
        z_resp(:,i) = x_resp(:,i*6);
    end
```

```
    semilogx(t_resp,-z_resp,'-b')
    drawnow
    hold on
    if max(max(z_resp)) > .99*d
        disp('Pullin at V_lo, decrease V_lo')
        go = 0;
    end
end

% find response at each voltage during iterative process to find Vpullin
while (V_hi-V_lo > 0.001 || max(max(z_resp)) < .99*d) && go == 1
    V_i = (V_hi+V_lo)/2
    [t_resp,x_resp,t_contact]                    =                  ode45(@(t,x)
dynamic_pullin_FACT(t,x,Tdot_0,Tddot_0,M,K,A_seff,d,V_i,epsilon_0),     tspan,
x_0,options);
    z_resp = zeros(size(x_resp,1),Nstages);
    disp(t_contact)
    for i = 1:Nstages
        z_resp(:,i) = x_resp(:,i*6);
    end
    semilogx(t_resp,-z_resp)
    drawnow
    hold on
    if max(max(z_resp)) > .99*d
        V_hi = V_i;
    else
        V_lo = V_i;
    end
end

% plot response of all stages' z DOF at Vpullin
figure(97)
semilogx(t_resp,-z_resp)
legend('Location','southwest')
xlabel('\itTime (s)')
ylabel('\itPosition (\mum)')
legend('Stage  1','Stage  2','Stage  3','Stage  4','Stage  5','Stage  6','Stage
7','Stage  8','Stage  9','Stage  10','Stage  11','Stage  12','Stage  13','Stage
14','Stage       15','Stage       16','Stage       17','Stage       18','Stage
19','Location','eastoutside')
axis([1e-11 1e-8 -1 0.2])
hold on

%% Natural Frequencies and Modes
[modes,omega_sq] = eig(M\K);
f_n = (omega_sq.^.5)./(2*pi);
f_n = diag(f_n);

%each column is a mode shape-- last row is its frequncy in Hz
f_n_modes = sortrows(cat(1,modes,f_n')',length(f_n)+1)';

%% Results Display

if  abs(t_contact - tlim)/tlim < 0.05
    disp('WARNING: Slow Pullin time- Consider increasing tlim')
end
```

```
disp(strcat("Pull-in at ",num2str(V_i),"V and ",num2str(t_contact),'sec'))
```

D2. MDOF Equations of Motion (dynamic_pullin_FACT.m)

This function is the MDOF analog to 'MechResp.m' in the SDOF section. It contains the equations of motion that govern the degrees of freedom of each stage in the ribbon.

```
function dxdt = dynamic_pullin_FACT(t,x,Tdot_0,Tddot_0,M,K,A,d,V,epsilon_0)
%Define number of possible degrees of freedom in system
n_DOF = length(Tdot_0);

%Definition of pull-in load wrench vector
W = zeros(n_DOF,1);
    for i = 1:n_DOF/6
        W(i*6-3) = (epsilon_0*A(i)*V^2) / (2*(d-x(6*i))^2);
    end
%Define derivatives to be used in
dxdt = [Tdot_0
        Tddot_0];
dxdt(1:n_DOF) = x(n_DOF+1:n_DOF*2);
dxdt(n_DOF+1:n_DOF*2) = M\(W - K * x(1:n_DOF));
```

D3. MDOF Pull-in Contact Trigger (dischargeContact.m)

This function is the MDOF analog to the function of the same name in the SDOF section. It indicates the stopping conditions for the ode45 function in the 'Vpullin_calculator_FACT.m' script.

```
%-------------------------------------------------------------------------
% This function defines the event for stopping dynamic_pullin_FACT.m, indicating
contact
% at the bottom of the trench and electrostatic discharge
%-------------------------------------------------------------------------

function [value,isterminal,direction] = dischargeContact(t,x,d)
Nstages = size(x,1)/12;
z_current = zeros(Nstages,1);
for i = 1:Nstages
    z_current(i) = x(i*6);
end
value = .9999*d-max(z_current); % depth  - displacement
isterminal = 1; % stop the integration
direction = 0; % any direction
end
```

D4. Mass Matrix Generator for Dynamic FACT (MassMatrix.m)

This function is modeled after the 'EulerStiffnessMatrix.m' function. It requires the stage parameters in a system of flexures to be specified in an input matrix as detailed in the function comments. The output is a matrix that can be used with an Euler stiffness matrix to generate the equations of motion of a flexure system as seen in Equation 10.

```
function [MM] = MassMatrix(StageParams)
%Function reads in a matrix (StageParams) that contains all the info needed
%to describe the stages in a flexure system, and returns its mass matrix

%Before you construct the matrix StageParams, be sure to number all of your
%system's rigid stages and number all of your flexure elements.  Grounded
%stages are numbered zero.

%Each row of the StageParams matrix corresponds to a rigid stage, the first
%row corresponding to stage one, the second to stage two (if it exists) and
%so on. There is no row for the grounded stage zero

%The following describes what each component of an individual row entails.
%the first 3 components are the chosen location vector, L, that points from
%the origin to the stage's center of mass
%the next 3 components are the direction of the n3 unit vector that points
%along the stage's Z'-axis.
%the next 3 components are the direction of the n2 unit vector that points
%along the stage's Y'-axis (perpendicular to l)
%the 10th component is the length of the element, l.
%the 11th component is the width of the element, b.
%the 12th component is the thickness of the element, h.
%the 13th component is the density of the material, rho.

format long;
N_stages = size(StageParams,1);
MM = zeros(6*N_stages);

%Read in paramters from StageParams input matrix
for i = 1:size(StageParams,1)
    L = StageParams(i,1:3);
    n3 = StageParams(i,4:6);
    n2 = StageParams(i,7:9);
    n1 = cross(n2,n3);
    l = StageParams(i,10);
    b = StageParams(i,11);
    h = StageParams(i,12);
    vol = l*b*h;
    rho = StageParams(i,13);

    Ix = ((rho*vol)/12) * (b^2+h^2);
    Iy = ((rho*vol)/12) * (l^2+h^2);
    Iz = ((rho*vol)/12) * (b^2+l^2);
```

```
    N =     [
            n1.'           n2.'           n3.'           zeros(3,1) zeros(3,1)
zeros(3,1)
            cross(L,n1).' cross(L,n2).' cross(L,n3).' n1.'      n2.'      n3.'
            ];
    DELTA =     [
                zeros(3)    eye(3)
                eye(3)      zeros(3)
                ];
    In = diag([Ix Iy Iz rho*vol rho*vol rho*vol]);
    M = N * DELTA * In / N;
    MM(6*i-5:6*i,6*i-5:6*i) = M;
end
```

References

1. Geim, A. K. & Novoselov, K. S. the Rise of Graphene. *Nat. Mater.* **6,** 1–14 (2007).
2. Singh, V. *et al*. Graphene based materials: Past, present and future. *Prog. Mater. Sci.* **56,** 1178–1271 (2011).
3. Grantab, R., Shenoy, V. B. & Ruoff, R. S. Anomalous Strength Characteristics of Tilt Grain Boundaries in Graphene. *Science (80-.).* **330,** 946–948 (2010).
4. Huang, P. Y. *et al*. Grains and grain boundaries in single-layer graphene atomic patchwork quilts. *Nature* **469,** 389–392 (2011).
5. Plimpton, S. J. LAMMPS Molecular Dynamics Simulator. *J. Comp. Phys* 1–19 (1995). doi:https://doi.org/10.1006/jcph.1995.1039
6. Stukowski, A. Visualization and analysis of atomistic simulation data with OVITO-the Open Visualization Tool. *Model. Simul. Mater. Sci. Eng.* **18,** (2010).
7. Kotakoski, J. & Meyer, J. C. Mechanical properties of polycrystalline graphene based on a realistic atomistic model. *Phys. Rev. B - Condens. Matter Mater. Phys.* **85,** 1–6 (2012).
8. Huang, Y., Wu, J. & Hwang, K. C. Thickness of graphene and single-wall carbon nanotubes. *Phys. Rev. B - Condens. Matter Mater. Phys.* **74,** 1–9 (2006).
9. Paradee, G. Fatigue Properties of Graphene Interconnects on Flexible Substrates. **15,** 423–428 (2014).
10. Cooper, D. R. *et al*. Experimental review of graphene. (2011). doi:10.5402/2012/501686
11. Isacsson, A. *et al*. Scaling properties of polycrystalline graphene: A review. *2D Mater.* **4,** (2017).
12. Qi, Z., Campbell, D. K. & Park, H. S. Atomistic simulations of tension-induced large

deformation and stretchability in graphene kirigami. *Phys. Rev. B - Condens. Matter Mater. Phys.* **90,** (2014).

13. Blees, M. K. *et al*. Graphene kirigami. *Nature* (2015). doi:10.1038/nature14588
14. Rokni, H. & Lu, W. Effect of graphene layers on static pull-in behavior of bilayer graphene/substrate electrostatic microactuators. *J. Microelectromechanical Syst.* **22,** 553–559 (2013).
15. Kwon, O. K., Lee, J. H., Park, J., Kim, K.-S. & Kang, J. W. Molecular dynamics simulation study on graphene-nanoribbon-resonators tuned by adjusting axial strain. *Curr. Appl. Phys.* **13,** 360–365 (2013).
16. Chen, C. *et al*. Graphene mechanical oscillators with tunable frequency. *Nat. Nanotechnol.* **8,** 923–927 (2013).
17. Bunch, J. S. *et al*. Electromechanical Resonators from Graphene Sheets. *Science (80-.).* (2007). doi:10.1126/science.1136836
18. She, Y. *et al*. The Effect of Viscous Air Damping on an Optically Actuated Multilayer MoS2 Nanomechanical Resonator Using Fabry-Perot Interference. *Nanomaterials* **6,** 162 (2016).
19. Ng, J., Chen, Q., Xie, Y.-H., Wang, A. & Wu, T. Comparative study between the fracture stress of poly-and single-crystalline graphene using a novel nanoelectromechanical system structure. *Micro Nano Lett.* **12,** 907–912 (2017).
20. *International Technology Roadmap for Semiconductors 2.0: Executive Report*. *International technology roadmap for semiconductors* (2015).
21. *White Paper 4: Understanding Electrical Overstress - EOS*. (2016).
22. Nathanson, H. C., Newell, W. E., Wickstrom, R. A. & Davis, J. R. The Resonant Gate Transistor. *IEEE Trans. Electron Devices* **ED-14,** 117–133 (1967).

23. Budynas, R. & Nisbett, K. Shigley's Mechanical Engineering Design, Ninth Edition. *McGraw Hill Inc.* 1–15 (2009).

24. Conley, H., Lavrik, N. V., Prasai, D. & Bolotin, K. I. Graphene bimetallic-like cantilevers: Probing graphene/substrate interactions. *Nano Lett.* **11,** 4748–4752 (2011).

25. Jacobsen, J. O., Winder, B. G., Howell, L. L. & Magleby, S. P. Lamina Emergent Mechanisms and Their Basic Elements. *J. Mech. Robot.* **2,** 011003–011003 (2009).

26. Hopkins, J. B. & Culpepper, M. L. Synthesis of precision serial flexure systems using freedom and constraint topologies (FACT). *Precis. Eng.* **35,** 638–649 (2011).

27. Howell, L. L. *Compliant Mechanisms*. (John Wiley & Sons, Inc., 2001).

28. *Graphene: Scientific background on the Nobel Prize in Physics 2010*. *The Royal Swedish Academy of Sciences* **50005,** (2010).

29. Rao, S. S. *Mechanical Vibrations*. *Recherche* **67,** (2010).

30. Hopkins, J. B. & Culpepper, M. L. Synthesis of multi-degree of freedom, parallel flexure system concepts via Freedom and Constraint Topology (FACT) - Part I: Principles. *Precis. Eng.* **34,** 259–270 (2010).

31. Hopkins, J. B. & Culpepper, M. Design of Parallel Flexure Systems via Freedom and Constraint Topologies (FACT). *Dep. Mech. Eng.* **MSc,** 393 (2007).

32. Wang, W., Shen, C., Li, S., Min, J. & Yi, C. Mechanical properties of single layer graphene nanoribbons through bending experimental simulations. *AIP Adv.* **4,** (2014).

33. Korhonen, T. & Koskinen, P. Limits of stability in supported graphene nanoribbons subject to bending. *Phys. Rev. B* (2016). doi:10.1103/PhysRevB.93.245405

34. Jiang, J.-W. & Park, H. S. Negative Poisson's Ratio in Single-Layer Graphene Ribbons. (2016). doi:10.1021/acs.nanolett.6b00311

35. Zandiatashbar, A. *et al*. Effect of defects on the intrinsic strength and stiffness of graphene. *Nat. Commun.* **5,** (2014).

36. Zhang, Y. Y., Pei, Q. X., Mai, Y. W. & Gu, Y. T. Temperature and strain-rate dependent fracture strength of graphynes. *J. Phys. D. Appl. Phys.* **47,** (2014).

37. Wang, S., Yang, B., Yuan, J., Si, Y. & Chen, H. Large-Scale Molecular Simulations on the Mechanical Response and Failure Behavior of a defective Graphene: Cases of 5–8–5 Defects. *Sci. Rep.* **5,** 14957 (2015).

38. Pei, Q. X., Zhang, Y. W. & Shenoy, V. B. A molecular dynamics study of the mechanical properties of hydrogen functionalized graphene. *Carbon N. Y.* **48,** 898–904 (2010).

39. Wilmes, A. A. R. & Renaud, A. A. Development of a multi-physics molecular dynamics finite element method for the virtual engineering design of nano-structures. (2016).

40. Wang, S. *et al*. Mechanical Properties and Failure Mechanisms of Graphene under a Central Load. *ChemPhysChem* **15,** 2749–2755 (2014).

41. Payne, C. M. *et al*. MOLECULAR DYNAMICS SIMULATION OF A NANOSCALE DEVICE FOR FAST SEQUENCING OF DNA. (2007).

42. Pun, G. P. P. & Mishin, Y. Optimized interatomic potential for silicon and its application to thermal stability of silicene. *Phys. Rev. B* **95,** 1–46 (2017).

43. Jhon, Y. I., Chung, P. S., Smith, R. & Jhon, M. S. Molecular Simulation of Fracture Dynamics of Symmetric Tilt Grain Boundaries in Graphene. 24 (2012).

44. Wei, Y. *et al*. The nature of strength enhancement and weakening by pentagon--heptagon defects in graphene. *Nat. Mater.* **11,** (2012).

45. Inui, N. & Iwasaki, S. Interaction Energy between Graphene and a Silicon Substrate Using Pairwise Summation of the Lennard-Jones Potential. *e-Journal Surf. Sci. Nanotechnol.* **15,**

40–49 (2017).

46. Artz, B. E. & Cathey, L. W. A finite element method for determining structural displacements resulting from electrostatic forces. in *Solid-State Sensor and Actuator Workshop* (IEEE, 1992). doi:https://doi.org/10.1109/SOLSEN.1992.228295

47. Ng, J. *et al*. Optimization of suspended graphene NEMS devices for electrostatic discharge applications. *2017 IEEE 12th Int. Conf. Nano/Micro Eng. Mol. Syst. NEMS 2017* 364–369 (2017). doi:10.1109/NEMS.2017.8017043

48. Strong, F. W., Skinner, J. L., Dentinger, P. M. & Tien, N. C. Electrical breakdown across micron scale gaps in MEMS structures. 611103 (2006). doi:10.1117/12.646508

49. Lee, C., Wei, X., Kysar, J. W. & Hone, J. Measurement of the Elastic Properties and Intrinsic Strength of Monolayer Graphene. *Science (80-.).* **321,** 385–388 (2008).