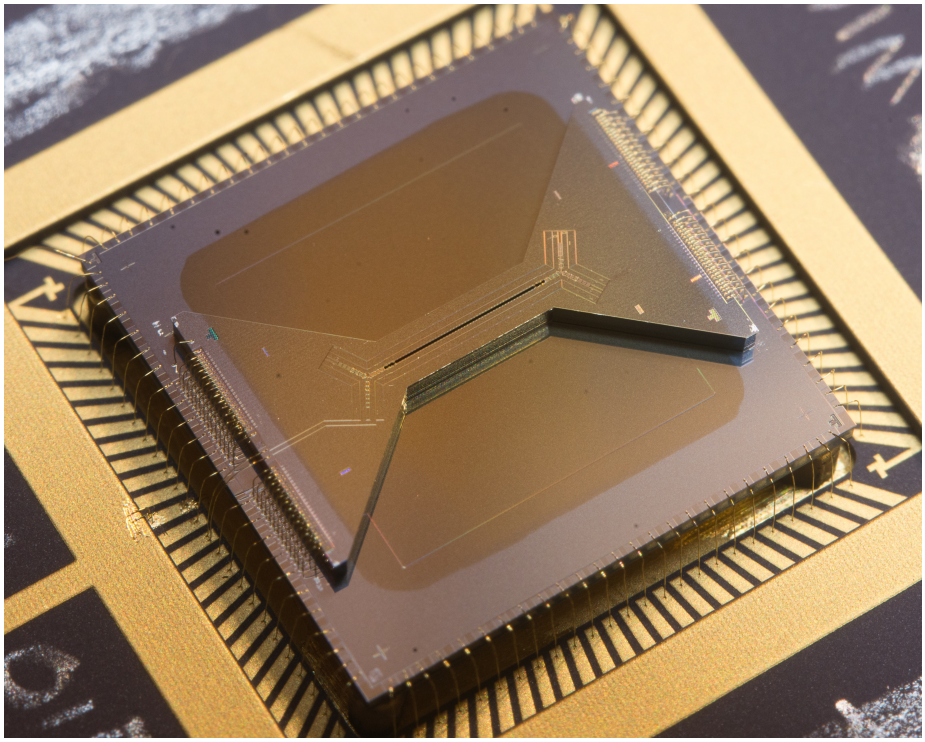
SAND2016-0796R

High Optical Access Trap 2.0



Version 1, January 26, 2016

Contact:
Peter Maunz
Sandia National Laboratories
Building 858EL/3301; MS 1082
1515 Eubank Blvd SE
Albuquerque, NM 87123
plmaunz@sandia.gov
505-844-8368

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energys National Nuclear Security Administration under contract DE-AC04-94AL85000.

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) under the Multi Qubit Coherent Operations (MQCO) program.

Revision	**Date Released**	**Comments**
1.0	January, 20$^{\text{th}}$, 2016	Initial version

Contents

1 Design Objectives

The High Optical Access (HOA) trap was designed in collaboration with the Modular Universal Scalable Ion-trap Quantum Computer (MUSIQC) team, funded along with Sandia National Laboratories through IARPA's Multi Qubit Coherent Operations (MQCO) program. The design of version 1 of the HOA trap was completed in September 2012 and initial devices were completed and packaged in February 2013. The second version of the High Optical Access Trap (HOA-2) was completed in September 2014 and is available at IARPA's disposal. The HOA trap was designed with the following design objectives:

1. High Optical Access for across chip laser beams, specifically the ability to focus a 370 nm beam to a 4 μm waist at the ion.
2. A slotted linear trap section that allows one to focus a 370 nm laser beam on the ion perpendicular to the trap surface with a waist of 2 μm.
3. Radial secular frequencies > 2 MHz for Yb with $< 300\,\mathrm{V_{amp}}$ of rf applied.
4. Full control of principal axes rotation.
5. Compatibility with a trench capacitor interposer.
6. Good axial voltage efficacy to enable the separation and recombination of ion chains.
7. Junctions for re-ordering ions with the ability to co-shuttle ions with mass ratio up to 5:4.

The first two design criteria were considered and optimized simultaneously for the slotted region of the trap (labeled "quantum" in Figure 1). The separate metrics generated competing design influences, since increasing ion height improves optical access but leads to a weaker trap for a given rf voltage. The decision to use a bow-tie platform was made because it provides high optical access for all laser directions between 45° and 135° relative to the trap axis, while leaving enough space on the trap ends to allow for wirebonding. A total of 94 control electrodes were required to accommodate the two junctions and a sufficiently long linear section of the trap connecting them.

1.1 Optical Access

The High Optical Access trap was designed to accommodate tightly focused laser beams across the surface or through the central slot of the trap in order to achieve high laser intensities at the ion positions and to enable individual addressing of ions in a chain. The optical access is parameterized by the fraction of a focused Gaussian beam which clips on a portion of the trap. In the case of the HOA trap, the trap itself, the interposer, and the package all impose similar restrictions, though ultimately the trap itself is the limitation when the isthmus width (the width of the bow-tie at its narrowest) exceeds 1.2 mm. The

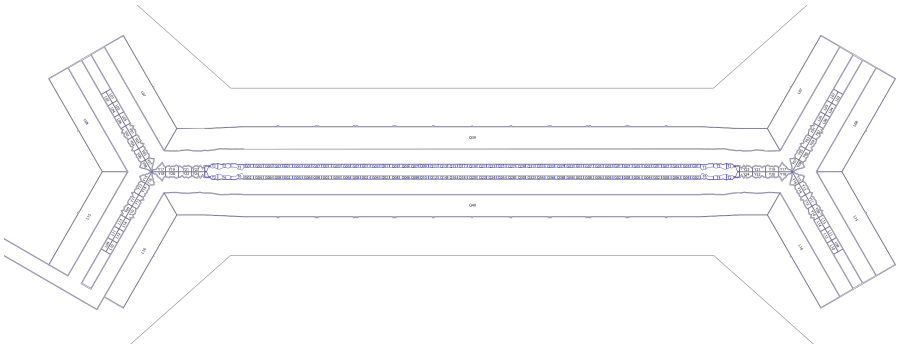


Figure 1: Schematic of the HOA-2 trap. The trap consists of a central slotted linear "quantum" section with 19 individually controllable electrode pairs. On either side of the quantum region is a shuttling region. Eight voltages control the 14 electrode pairs on either side of the quantum region and allow shuttling. The transitions between slotted and un-slotted region, junctions and loading regions on the left and right side are co-wired to stay within the available number of 94 control voltages. The schematic con also be found as an attached file `HOA2-RS1096.pdf` in scale 25.4:1 (1" corresponds to 1 mm). Distances can be measured using the Measuring Tool in Adobe Reader.

numerical aperture for a beam skimming the surface perpendicular to the trap axis is 0.11, for a beam at 45° a numerical aperture of 0.08 is available. In Figure 2 the clearance of a horizontal laser beam perpendicular to the trap axis across the 1.2 mm isthmus of the HOA-2 trap is analyzed. The graph is a plot of the distance from the beam center to the isthmus edge (in multiples of the beam radius). The interposer, CPGA package, and isthmus were all analyzed for 370 nm light directed across the HOA-2 trap. Since we were interested in focal spots with waist radii $< 2.5\,\mu$m we chose an isthmus width commensurate with the limitations imposed by the CPGA and interposer. A plot of the 8.8 mm die (such as the linear Thunderbird trap) is included for comparison.

In addition to being concerned about achieving high optical access for a side laser, we also needed to preserve high optical access through the slot. Figure 3 shows the beam clearance analysis in the slotted region of the trap. For a beam perpendicular to the chip, the available numerical aperture is 0.25.

1.2 rf Pseudo-potential optimization

The pseudo-potential strength of the trap was maximized by optimizing electrode configurations (rf electrode widths, gaps, realization of inner control electrodes on a lower metal layer) within fabrication constraints and maintaining a minimum ion height of 70 μm. Ultimately the optimization converged to place the ion 68 μm above the top metal surface. Figure 16 shows a cross-section of the slotted region and the electrode widths and layers that were used to

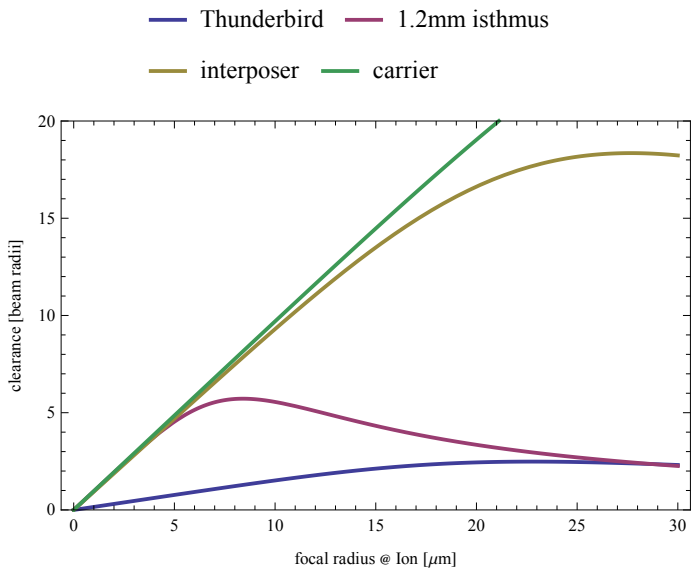


Figure 2: Clearance for the surface (horizontal) laser beams, analyzed for the trap, interposer, and CPGA package, assuming a 369 nm laser. A distance to beam radius ratio of 2.5 corresponds $< 3 \times 10^{-6}$ of the power clipping, while a ratio of 4 corresponds to $< 2 \times 10^{-14}$ of the power of an ideal Gaussian beam clipping on the device. The available numerical aperture for a beam skimming the surface is 0.11 and 0.08 if the beam has an angle of 90° and 45° with respect to the trap axis, respectively.

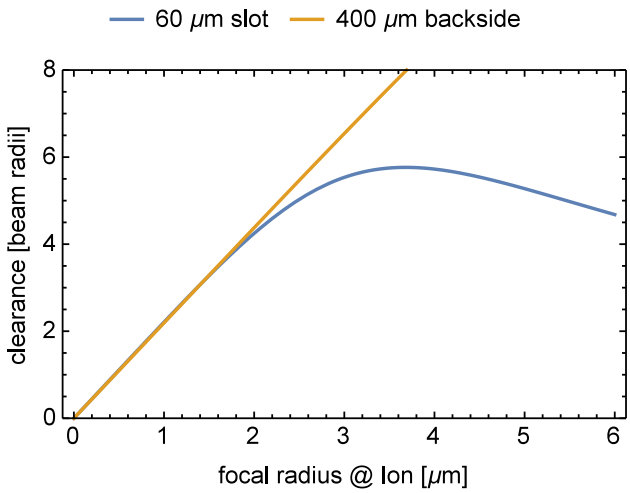


Figure 3: Clearance for a vertical laser beam through the 60 μm slot of the HOA-2 trap, analyzed for the slot (corresponding to the top silicon directly below the trap metal) and the backside slot (400 μm full width through the bottom of the silicon substrate). For a beam through the slot a numerical aperture of 0.25 is available.

maximize trap strength. Compared to previous SNL fabricated traps, the most important parameters for increasing the trap strength at a given rf voltage were increasing the rf electrode width and exposing the rf grounded M3 level inside the rf electrodes (see Figures 7 and 16).

1.3 Trap topology

A scanning electron micro-graph (SEM) and schematic of the HOA trap are shown in Figures 5 and 1, respectively. The device is broken out into 4 types of regions:

1. a "quantum" region which is useful for transverse gates because of the slot underneath the trap and the strong trapping potential;
2. a "transition" region which connects the slotted and un-slotted parts of the trap;
3. a "junction" region which can be used for re-ordering ions in the quantum region;
4. and a "loading" region which has a loading hole for backside ion loading. Note that the slotted quantum region can also be used for loading if convenient.

Control signal	axial location	Control signal	axial location
Q19, Q20	$0\,\mu$m	Q19, Q20	$0\,\mu$m
Q17, Q18	$-70\,\mu$m	Q21, Q22	$70\,\mu$m
Q15, Q16	$-140\,\mu$m	Q23, Q24	$140\,\mu$m
Q13, Q14	$-210\,\mu$m	Q25, Q26	$210\,\mu$m
Q11, Q12	$-280\,\mu$m	Q27, Q28	$280\,\mu$m
Q09, Q10	$-350\,\mu$m	Q29, Q30	$350\,\mu$m
Q07, Q08	$-420\,\mu$m	Q31, Q32	$420\,\mu$m
Q05, Q06	$-490\,\mu$m	Q33, Q34	$490\,\mu$m
Q03, Q04	$-560\,\mu$m	Q35, Q36	$560\,\mu$m
Q01, Q02	$-630\,\mu$m	Q37, Q38	$630\,\mu$m
G03, G04	$-700\,\mu$m	G03, G04	$700\,\mu$m
G01, G02	$-770\,\mu$m	G01, G02	$770\,\mu$m
G07, G08	$-840\,\mu$m	G07, G08	$840\,\mu$m
G05, G06	$-910\,\mu$m	G05, G06	$910\,\mu$m
G03, G04	$-980\,\mu$m	G03, G04	$980\,\mu$m
G01, G02	$-1050\,\mu$m	G01, G02	$1050\,\mu$m
G07, G08	$-1120\,\mu$m	G07, G08	$1120\,\mu$m
G05, G06	$-1190\,\mu$m	G05, G06	$1190\,\mu$m
G03, G04	$-1260\,\mu$m	G03, G04	$1260\,\mu$m
G01, G02	$-1330\,\mu$m	G01, G02	$1330\,\mu$m
G07, G08	$-1400\,\mu$m	G07, G08	$1400\,\mu$m
G05, G06	$-1470\,\mu$m	G05, G06	$1470\,\mu$m
G03, G04	$-1540\,\mu$m	G03, G04	$1540\,\mu$m
G01, G02	$-1610\,\mu$m	G01, G02	$1610\,\mu$m
T5, T6	$-1692\,\mu$m	T5, T6	$1692\,\mu$m
T3, T4	$-1782\,\mu$m	T3, T4	$1782\,\mu$m
T1, T2	$-1880\,\mu$m	T1, T2	$1880\,\mu$m
Y23, Y24	$-1975.2\,\mu$m	Y23, Y24	$1975.2\,\mu$m
Y21, Y22	$-2069.6\,\mu$m	Y21, Y22	$2069.6\,\mu$m
Y19, Y20	$-2163.9\,\mu$m	Y19, Y20	$2163.9\,\mu$m
Y17, Y18	$-2259.1\,\mu$m	Y17, Y18	$2259.1\,\mu$m
junction center	$-2308.5\,\mu$m	junction center	$2308.5\,\mu$m

Table 1: Axial location of the centers of all electrodes on the central linear section, transitions and junctions. The electrode pitch in the Quantum and Shuttling regions is $70\,\mu$m.

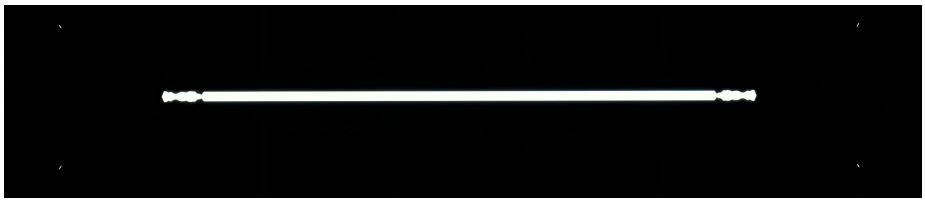


Figure 4: Optical picture of the back illuminated trap. The central slot as well as the four loading holes appear bright.

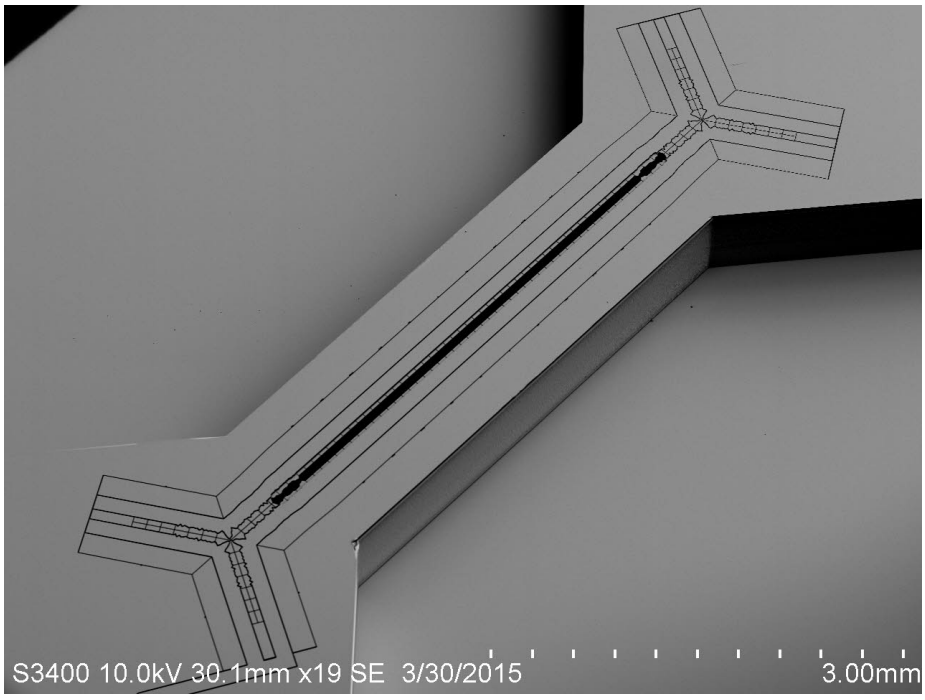


Figure 5: Scanning electron micrograph of the HOA trap.

We define the coordinate system with the x-axis in the plane of the top metal level along the long linear region of the trap, the y-axis in the plane of the trap surface perpendicular to the linear axis of the trap, and the z-axis perpendicular to the trap surface. The origin of the coordinate system is at the center of symmetry of the trap, between electrodes Q19 and Q20. In the vertical direction $z = 0$ is defined as the top of M4. Because the inner control electrodes are located on M3, they are located at $z \approx -12\,\mu$m.

1.3.1 Quantum region

Figure 6: In the quantum region the electrode pitch is $70\,\mu$m, the central slot width is $60\,\mu$m, and the vertical offset between M4 electrodes and M3 electrodes is $\approx 11.8\,\mu$m (trap specific values can be found in the trap device specific documentation). The electrodes with blue contour are located on M3, electrodes with black contour are located on M4. Beam alignment marks can be found between the outer control electrodes and the outer ground plane. Alignment marks are centered at $0\,\mu$m (single triangle), $\pm 280\,\mu$m (double triangle), $\pm 560\,\mu$m (triple triangle), $\pm 840\,\mu$m (quadruple triangle), $\pm 1120\,\mu$m (triple triangle), and $\pm 1400\,\mu$m (double triangle).

In the 1.33 mm long central quantum region, 19 control electrode pairs with independent control voltages are available to generate axial trapping potentials (Figure 6). The electrode pairs have a pitch of $70\,\mu$m. The width of the central slot is $60\,\mu$m. The location of electrode pair centers along the x-axis is listed in Table 1.

1.4 Shuttling and transition regions

The linear section has many more electrodes than independent control voltages are available in the standard CPGA package. Therefore, it is necessary to use

the same voltages on multiple electrodes. While the central quantum region uses 19 independent pairs of inner control electrodes, the shuttling regions on either side of the quantum region only use 4 control electrode pairs that are repeated 3.5 times on either side (Figure 8). Thus, multiple trap minima can be moved through the shuttling region simultaneously.

The availability of a slotted region is important for individual addressing of ions in a chain using the high numerical aperture imaging optic used to image trapped ions. However, to make a design with slotted regions scalable transitions between slotted and above-surface regions are necessary. A naive design of this transition would lead to large pseudo-potential bumps. Modulations of the inner control electrodes as well as of the rf electrodes and outer control electrodes are used to reduce the height of the rf pseudo-potential bumps a naive design would have. The residual bump is below 1 meV for a 250 meV deep trap and is shown in Figure 9.

1.5 Junction and loading regions

The junction is modeled after the Y-junction realized in Sandia's earlier Y-junction trap [1, 2]. In contrast to the Y-junction trap, the HOA-2 trap uses segmentation of the inner control electrodes instead of the outer control electrodes. This increases the number of control electrodes and the available gradients of the electrostatic potentials in the junction and opens up the design space for shuttling ions through the junction.

The residual axial rf-pseudopotential in the junction is shown in Figure 9. The maximal height of the residual rf bumps is less than for the transition region. In Figure 11, the height of the rf nodal line above the top metal surface of the trap is depicted. It shows that the rf nodal height varies between 70 μm in the slotted region and 85 μm in the junction.

2 Fabrication and packaging

The delivered devices contain the following materials:

Device Silicon, silicon dioxide, silicon nitride, aluminum, copper, titanium nitride, titanium, platinum, tungsten, gold

Die attach EpoTek H21D, or SAC solder (as specified in the device specific documentation).

Package Alumina, gold, nickel, Alloy 42 (FeNi, 58%/42%); the data sheet `CPG10039.pdf` provides more detail.

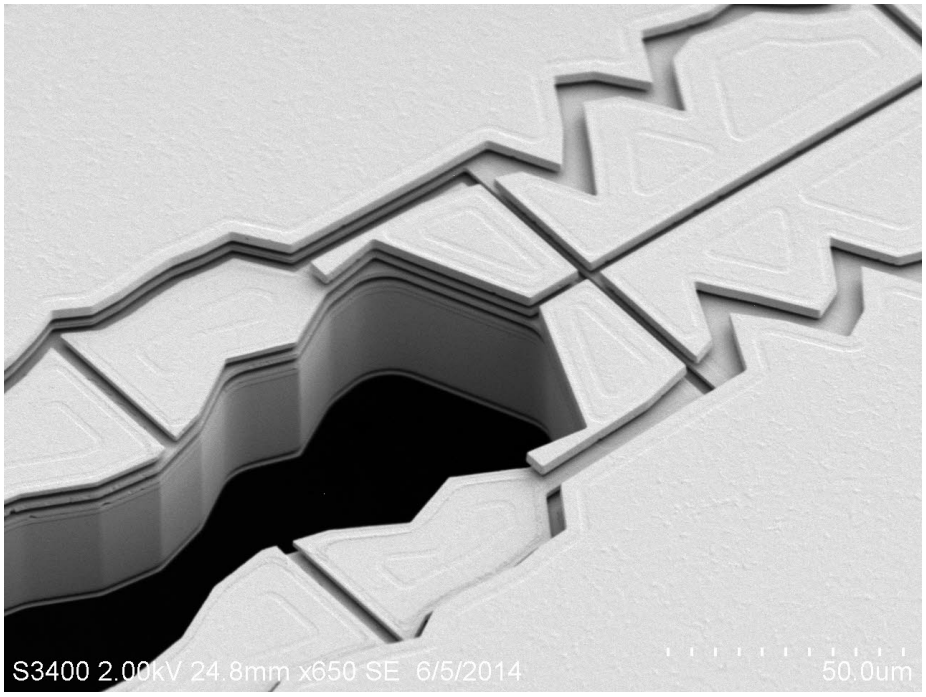


Figure 7: SEM micro-graph showing the transition between slotted and above-surface parts of the linear trap. In the above-surface part, the segmented inner control electrodes are located in the top metal layer (M4). In the slotted region, the inner control electrodes are located on the second metal layer from the top (M3). The modulation of electrode shapes is optimized to reduce rf pseudo potential bumps along the rf nodal line.

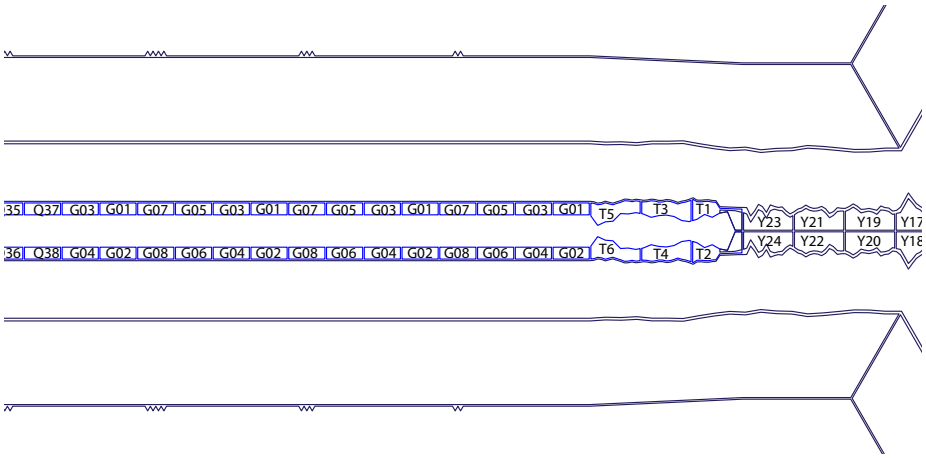


Figure 8: Shuttling and transition regions. In the shuttling and transition regions several electrodes are co-wired. Electrodes are labeled with the control signal name.

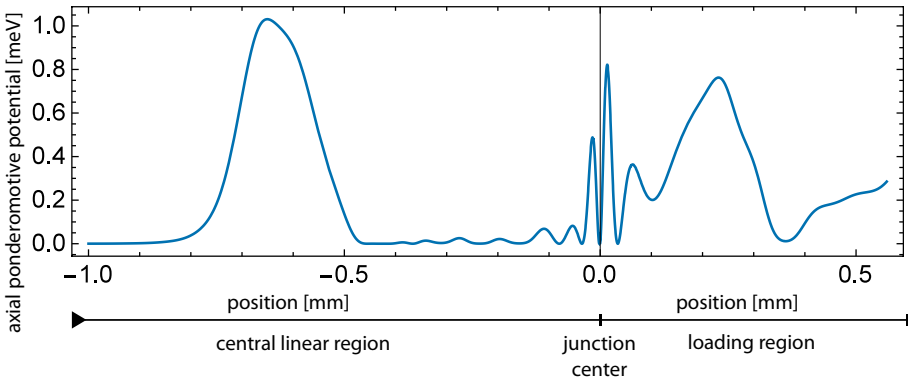


Figure 9: Axial ponderomotive potential (^{171}Yb, $\Omega_{\mathrm{rf}} = 2\pi\,\mathrm{MHz}$, $V_{\mathrm{rf}} = 250\,\mathrm{V}$) in the transition region and Y-junction along the nodal line of the rf potential. The projection of the nodal line of the trap is depicted in Figure 10. The potential hump between $-0.8\,\mathrm{mm}$ and $-0.5\,\mathrm{mm}$ is located at the transition between slotted and un-slotted trap regions, the center of the junction is at $0\,\mathrm{mm}$ and the loading hole is centered at $0.52\,\mathrm{mm}$.

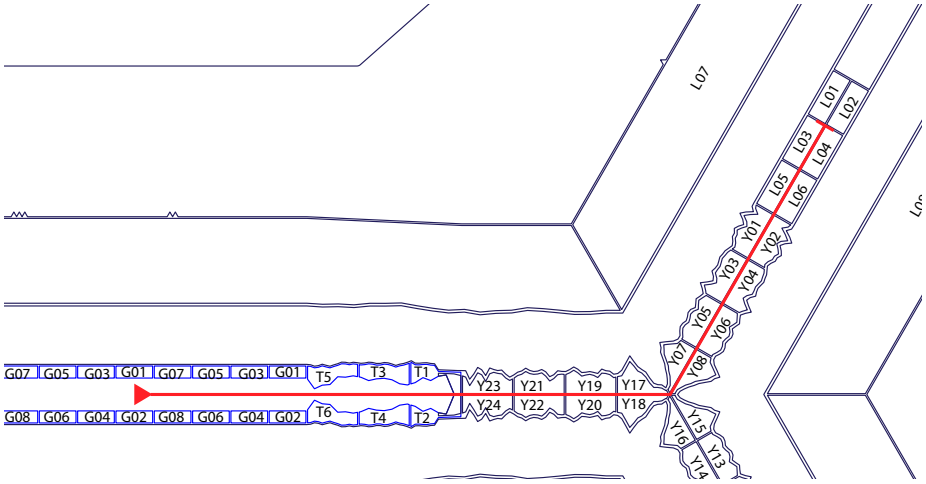


Figure 10: Schematic of the trajectory along which the data of Figures 9, 11 and 12 are taken.

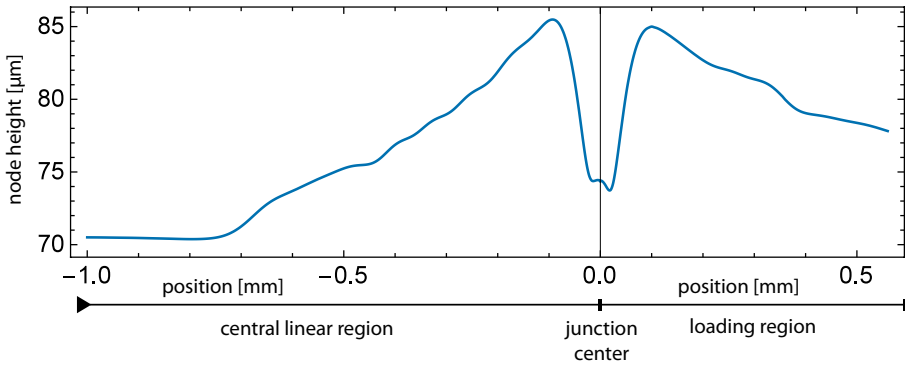


Figure 11: Height of the rf node above the top metal of the trap.

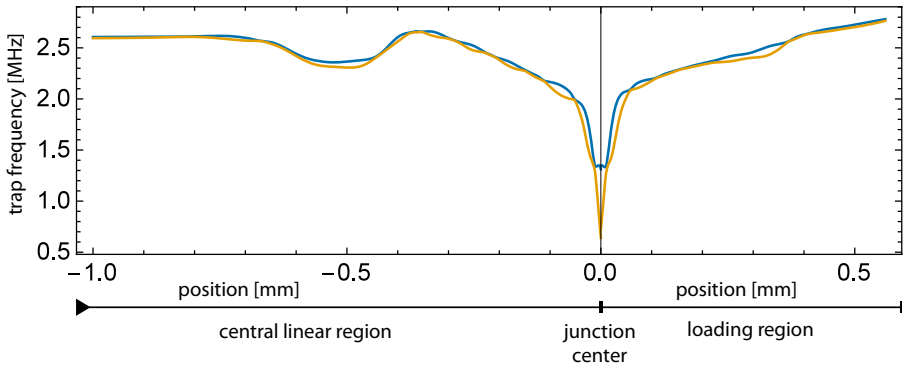


Figure 12: Radial trap frequencies along the path depicted in Figure 10 without principal axes rotation (^{171}Yb, $\Omega_{\mathrm{rf}} = 2\pi\,\mathrm{MHz}$ and $V_{\mathrm{rf}} = 250\,\mathrm{V}$). The trap frequencies parallel (blue) and perpendicular (yellow) to the trap surface reach a minimum in the center of the junction. The very low trap frequency in the junction center could be avoided by choosing a shuttling path that avoids the center of the junction at the expense of increased micro-motion [1].

2.1 Interposer

Control electrodes require capacitors to shunt to ground rf signals, which couple via the parasitic capacitance between the rf electrode and the control electrodes. This is achieved on the HOA 2.0 device using an interposer chip with a trench capacitor for each control channel. The interposer chip also provides a distribution layer such that bondpads which only exist along a few mm on two edges of the HOA 2.0 trap are re-routed to the entire perimeter (4 sides) of the interposer, to align with the package pads. A circuit diagram of the trench capacitors is shown in Figure 14.

Each capacitor has a capacitance $C = 1.05\,\mathrm{nF}$, with a variance $< 1\%$. The trench capacitors are designed to be operated at up to $\pm 20\,\mathrm{V}$ and have a breakdown voltage of $> \pm 30\,\mathrm{V}$. The stray inductance of the trench capacitors themselves, which can undermine the ability to shunt the rf off the control electrodes, is 0.05 nH, much smaller than the $\approx 1\,\mathrm{nH}$ inductance that the wirebonds between the trap and the interposer add. The series resistance of the trench capacitors ($4\,\Omega$) is comparable to the lead resistance of the routing between bondpad and electrode on the trap chip. In Figure 15 we calculate the voltage pickup on the DC electrodes for 200 V amplitude of rf applied.

2.2 Trap chip

These devices were fabricated using 4 metal levels (Figure 16); the top (M4) is the electrode level (although HOA 2 also uses the second metal level M3 for the inner control electrodes), the lower metal layers (M1, M2 and M3) are used for routing control lines. In locations where metal below the electrodes is

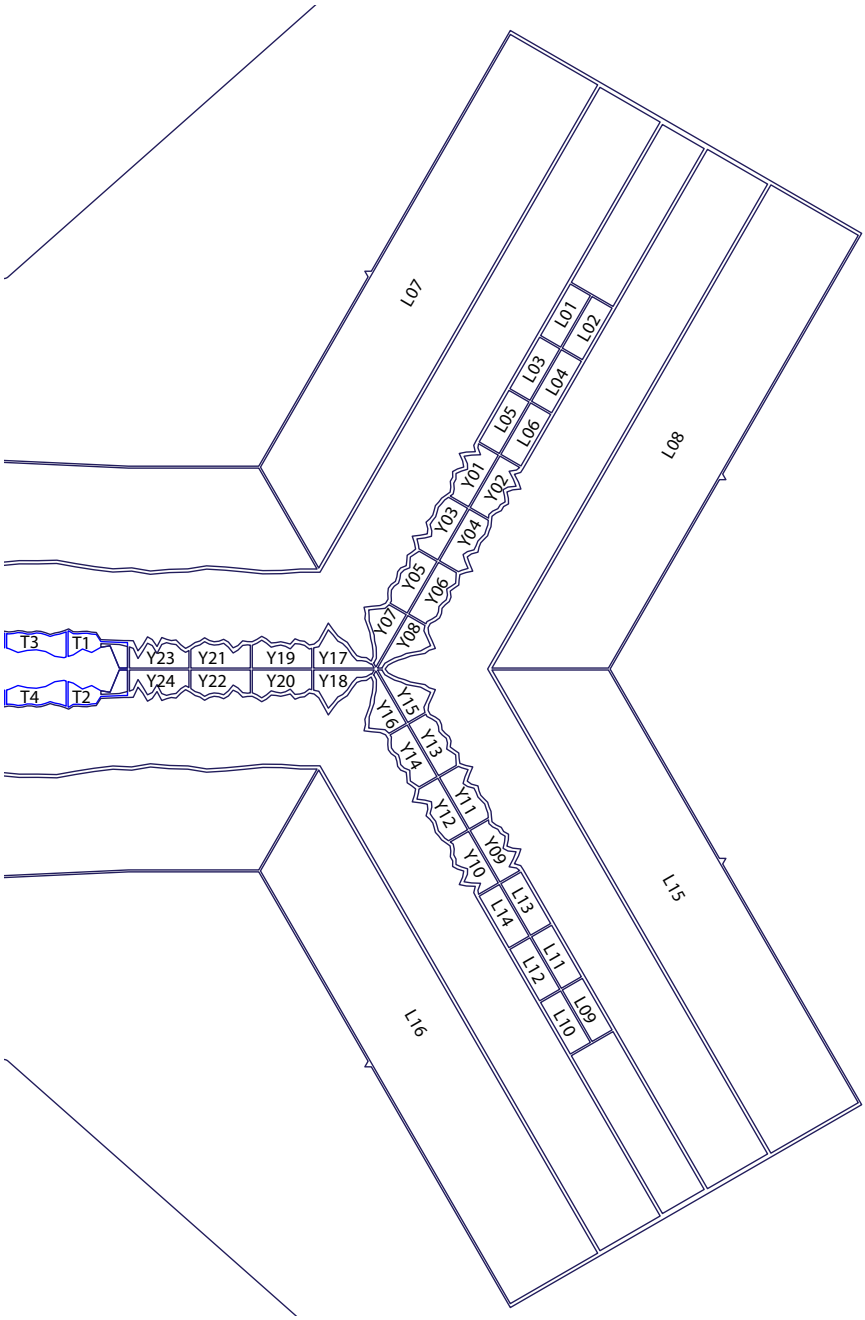


Figure 13: Junction and loading regions: The two junction and loading regions are co-wired. The loading arms are at an angle of $\pm 60^\circ$ with respect to the linear section.

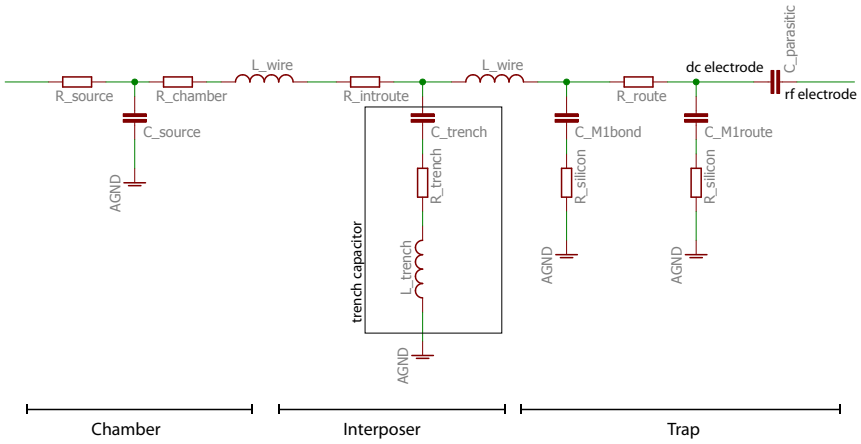


Figure 14: Trench capacitor circuit model.

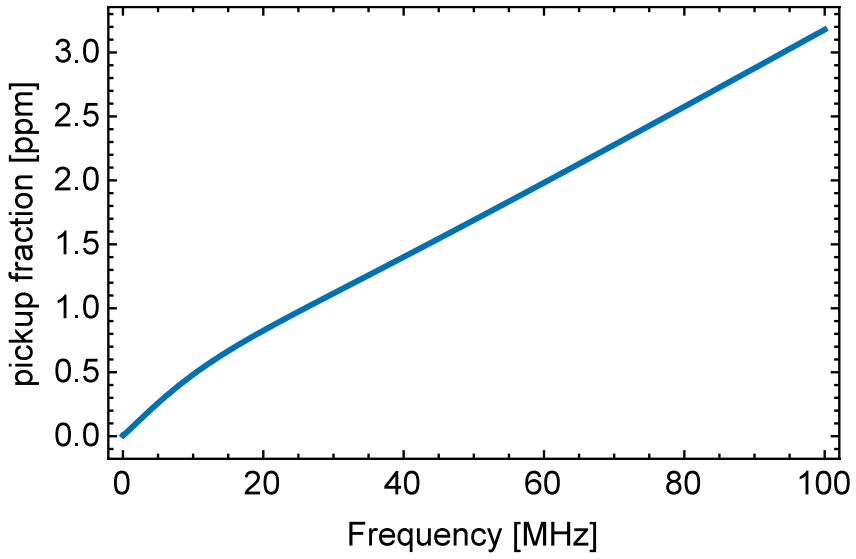


Figure 15: Calculated levels of pickup in parts per million (ppm). The model described in the text is evaluated for the following parameters: $C_{\mathrm{parasitic}} = 1\,\mathrm{fF}$, $R_{\mathrm{route}} = 2.6\,\Omega$, $C_{\mathrm{M1bond}} = 1.7\,\mathrm{pF}$, $R_{\mathrm{silicon}} = 4\,\Omega$, $L_{\mathrm{wire}} = 1\,\mathrm{nH}$, $C_{\mathrm{trench}} = 1\,\mathrm{nF}$, $L_{\mathrm{trench}} = 50\,\mathrm{pH}$, $R_{\mathrm{trench}} = 4\,\Omega$, $R_{\mathrm{introute}} = 4\,\Omega$, $R_{\mathrm{chamber}} = 2\,\Omega$, $C_{\mathrm{source}} = 100\,\mathrm{nF}$, $R_{\mathrm{source}} = 1\,\mathrm{k}\Omega$. According to this model the estimated pickup for an rf frequency of 100 MHz and rf voltage of 200 V remains below 1 mV.

exposed to the ion, the exposed metal is grounded. This applies for the central segmented control electrodes on M3 where M2 is grounded as well as for the M3 metal exposed through gaps in the M4 plane. AlCu (99.5%/0.5%) is used for the metal levels, with vias for vertical interconnections. All electrodes are overhung from the underlying silicon oxide insulating layers. Unless otherwise specified, the top metal is over-coated with 50 nm of gold, using Ti and Pt for adhesion.

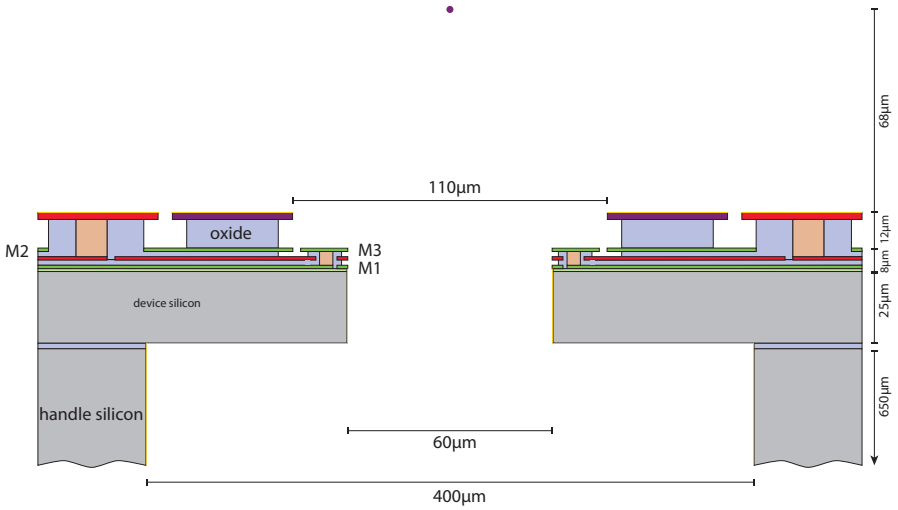


Figure 16: Cross section schematic of the slotted region.

2.3 Package and die attach

These devices are packaged on a High Temperature Co-fired Ceramic (HTCC) NTK package (see `CPG10039.pdf`). The trap and interposer are attached on top of a 1.52 mm thick alumina spacer on the package using EpoTek H21D (`H21D.pdf`) so that the device projects above the surface of the package. This die attach was chosen for its low outgassing properties and electrical and thermal conduction (because the die attach grounds the spacer to the package base). The die attach is cured at $225°C$ for a duration of two hours to ensure a complete cure. We recommend staying below $225°C$ for the vacuum bake because that is the maximum temperature we've exposed it to, but its glass transition temperature is $240°C$ and its glass melting point is $370°C$, so in principal it could go much higher.

The alumina spacer is coated with Ti/Pt/Au, with thicknesses of 1000Å / 1000Å / 5000Å. Alumina was chosen to match the thermal expansion properties of the package and the metal coatings provide a grounded metal to eliminate

charge buildup. The full package stack-up can be seen in Figure 17 (the top two components are the trap and interposer).

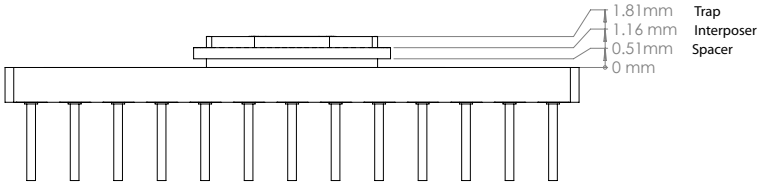


Figure 17: Package stack-up.

2.4 Wirebonding

Wirebond parameters are optimized to achieve a low profile, with $< 25\,\mu$m of projection above the surface of the trap chip (in addition to the thickness of the wire), using $25\,\mu$m round gold wire. The trap chip bond pads are located on the long ends of the bowtie and are $75\,\mu$m wide with a $7\,\mu$m gap to each neighbor. The control electrodes are bonded to the interposer and then the package, while the rf is wired (with 2 wires) directly to the package. This is to reduce the additional capacitance of the interposer, but it does not achieve the low profile characteristics of the control wirebonds. Figure 18 shows images of the low profile wirebonds.

2.5 Testing

Before delivery the packaged traps are tested for shorts between every combination of electrodes. The capacitance is measured from every control electrode to ground as a test of the interposer as well. Using a charge induced image contrast technique with an SEM the electrode connections are verified, testing the possibility that a disconnected wirebond or internal via leaves the electrode floating. The rf electrode resistance is measured to ground with a separate multi-meter to ensure that it exceeds $40\,\mathrm{M}\Omega$.

Finally, the device is inspected optically to ensure no surface contaminants are present on the trap which would interfere with trap performance.

The results of these tests are documented and shipped with the devices, which are labeled with a unique designator on the package surface that also provides internal information regarding the specific wafer and location of the die, as well as the photomasks used in fabrication. An example of this documentation can be seen in MT099002I00001A-W03-07.pptx. If any non-standard features are present (such as wirebond jumpers or additional capacitors), that will be detailed in the part-specific testing documentation.

The devices are maintained in a cleanroom environment throughout their assembly. After final electrical testing they are plasma cleaned, stored in a package and placed in an antistatic bag for shipment.

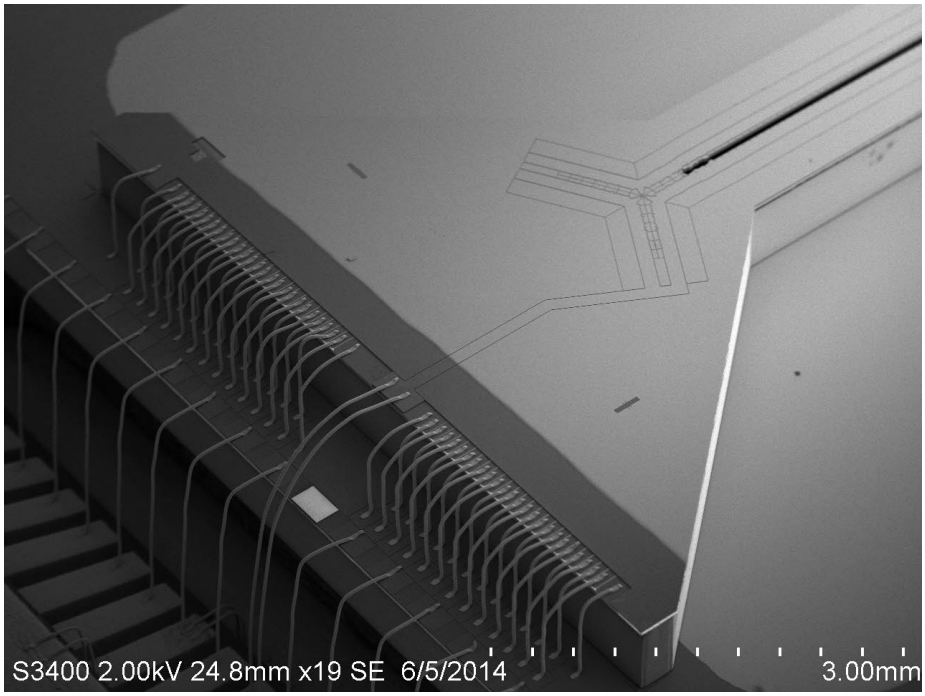


Figure 18: Scanning elecron micrograph of the wirebonds from the HOA 2 trap chip to the interposer. Wirebonds connect all dc control signals from the trap to the interposer where the trench capacitors are located underneath the bond pads. The rf electrode is connected with two wirebonds directly to the package to keep the capacitance as low as possible.

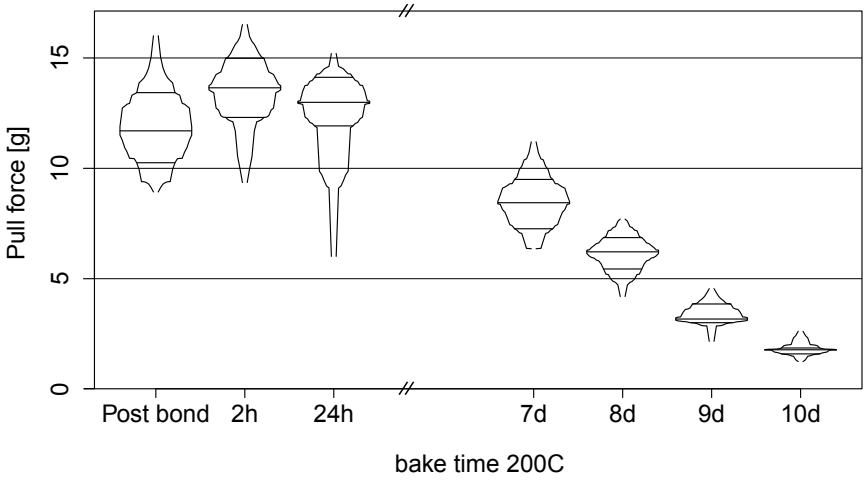


Figure 19: Wirebond pull force after bake. The box percentile graph shows the distribution and first, second and third quartile of wirebonding pull strength after bake in vacuum. Gold wires are bonded to aluminum pads. The inter-diffusion of gold and aluminum leads to the formation of brittle inter-metallic compounds (purple plague). After very long bakes (longer than tested) the pull strength of the wirebond can fall below the stress of the wire and the wirebond can detach. A pull strength of $> 5\,\mathrm{g}$ is sufficient to ensure that wirebonds do not detach. Therefore, the vacuum bake at $200\,^{\circ}\mathrm{C}$ should be restricted to less than 7 days.

3 Installation

The following describes the installation procedure for the user after receiving the trap.

Microfabricated surface traps are sensitive to dust accumulation on the surface. At Sandia, the traps are only handled in cleanrooms. The traps are packaged in a cleanroom and should *only be opened in a cleanroom.*

The HOA-2 trap with integrated trench capacitors can be damaged by static electricity. Voltages above $\pm 30\,\mathrm{V}$ will damage the trench capacitors attached to all control electrodes.

3.1 Vacuum chamber

The user is responsible for building a UHV vacuum chamber to accommodate the packaged device. A chamber with 94 control voltages wired to be compatible with the MQCO standard (see `packageStandard_v2.0.pdf`) is necessary for the successful operation of the trap.

The success of these experiments depends on achieving an excellent vacuum. All surfaces exposed to the interior of the chamber need to be handled in an absolutely grease-free way. Pressures below $10^{-10}\,\mathrm{mbar}$ should be acceptable for many ion trapping experiments, although pressures significantly lower will result in superior trapping times (since the trap depth is only a few times room temperature).

In addition the traps are sensitive to dust particles accumulated on the surface, as they can cause electrical shorts, accumulate static charges, and cause laser scatter. Therefore, we strongly recommend installing the traps in a cleanroom environment to prevent contamination.

The orientation of the trap in the vacuum chamber has an influence on the likelyhood of getting dust particles located on the trap surface. If the trap surface is vertical or pointing down, dust accumulation on the surface is less likely.

Vacuum compatible zero insertion force (ZIF) sockets for the 100 pin CPGA package are available from Tactic Electronics (PN: 100-4680-001A).

3.2 Ground plane above trap

The user is responsible for mitigating charge on nearby dielectric surfaces. Generally the closest and most influential surface is the imaging viewport. While the package itself is not much farther away from the ion, the geometry limits the impact of charge that may build up on that surface (since there are many screening grounded regions between exposed dielectric of the package and the ion). The viewport, however, has direct line-of-sight to the ion and unless mitigated is mostly unscreened. Possible approaches include:

1. Building a custom metal screen which accommodates the necessary laser and imaging access and mounting this structure on the package. Screens with openings up to an NA of 0.6 have been operated successfully.

2. Coating the re-entrant viewport with a conductive transparent material (such as ITO)

3. Using a mesh with a small geometric fill factor.

In addition the screen should be placed far enough away as to not have a significant impact on the electric field for an otherwise unscreened trap, which is how they are simulated. A rule of thumb is that if the distance of the screen to the trap surface is more than 20 times the distance of the ion above the trap surface (70 μm), then the presence of the ground will have minimal impact on the trap behavior. Above this separation, the specific distance does not significantly affect operating conditions, such as the trap strength for a given rf voltage or the control waveforms. A 5 mm distance between a grounded mesh and the trapping surface has worked well on experiments at SNL. If the screen is closer to the trap surface, the trap will still be functional, however, the voltage solutions have to be adjusted for the presence of the ground screen.

3.3 Trap insertion

These traps have to be installed with the gold tab on the package pointed in the direction of the socket's rf pins. The packages are quite robust and can be pushed in with a great deal of pressure, however the user should be careful not to touch the wirebonds on the trap, but only contact the package surface. Please also make sure that none of the pins is being bent while inserting the package into the socket. If the trap needs to be retracted and used again, care should be taken to not bend the package pins.

The traps has to be installed in a dust-free and clean environment, ideally a ceanroom, to limit the possibility of dust falling on the chip surface, but if that is not feasible they should be installed as quickly as possible to minimize exposure.

3.4 Pre-bake testing

Before baking the user should verify that the rf electrode is not shorted to ground and measure its capacitance. The capacitance of the packaged device should be 11 pF, 7 ± 1 pF for the trap die and 4 ± 1 pF for the CPGA carrier. This does not include the capacitance of the feedthrough and socket (which depends on the wiring, but is ≈ 6 pF in our experiment) and should be measured by the user before device insertion. If higher than normal voltages are to be applied, this should be tested at high vacuum before baking to make sure the device can handle them. Tests can be done with a DC voltage which should be applied through a large resistor to limit the current. The leaking current should be measured with a picoamp meter. In this setup, a rising current is indicative of a voltage close to breakdown.

The user should also verify that all DC control electrodes are not shorted to ground and measure their capacitance; if the capacitance is a multiple of 1 nF,

the electrode is likely shorted to a neighbor (either on the chip or in the wiring of the vacuum chamber).

3.5 Baking

The devices themselves can be baked up to $200°C$, limited by the die attach and the development of purple plague [3]. It may be the case that other components (such as Kapton wires) in the vacuum chamber limit the bake to a lower temperature.

As described in section 2.4, the device is packaged using wirebonding with gold wires connecting to aluminum pads. If this junction is heated, brittle aluminum-gold intermetallics are formed. The aluminum-gold intermetallic will lead to a slightly higher contact resistance and, most importantly, to a reduced pull strength of the wirebonds. This process depends strongly on the baking temperature (see Figure 2 in [3]), thus limiting the baking time at high temperatures is most effective in preventing the formation of purple plague. Our tests (Figure 19) show that baking of up to 7 days at 200 °C is acceptable.

4 Operational specifications

4.1 rf voltage application

The HOA devices have a total capacitance between rf hi and ground of 11 pF, including the package (which adds 4 pF). Depending on the vacuum chamber, the feedthrough, internal rf wiring, and socket can add an additional ≈ 6 pF, bringing the total capacitive load to ≈ 17 pF.

In an ideal quadrupole trap, the trap secular frequency is given by:

$$\omega_{\text{sec}} = \frac{qV}{\sqrt{2}\Omega_{\text{rf}}\, mR^2},$$

where q is the charge of the trapped particle, V is the voltage amplitude, Ω_{rf} is the applied rf angular frequency, m is the mass of the particle, and R is electrode distance. This can be simply modified to calculate the secular frequency in the HOA trap by substituting R for the characteristic distance of the device. The characteristic distance corresponds to the distance at which hyperbolic electrodes operating at the same voltage and frequency would generate the same secular frequency as in the surface trap. The characteristic distance for the HOA 2 trap in the slotted region is

$$R = 140\,\mu\text{m}\,.$$

By substituting R with this number the user can calculate the secular frequency they are generating for a particular voltage, drive frequency, and ion species.

The trap depth can be expressed as:

$$d = \alpha\frac{1}{2}m\omega_{\text{sec}}^2 R^2, \quad \alpha = 0.028,$$

where the factor α is required to account for the specifics of the surface trap geometry. In terms of practical trap depth (defined as the ability to stay trapped following a collision with a particle of energy E, this assumes that the stability parameter h does not exceed 0.5.

These devices have been safely operated with rf amplitudes of up to 300 V, but could possibly be operated at higher voltages if necessary. This would have to be tested by the user.

4.2 Control voltages

Control voltages up to ±20 V are safe. The HOA trap interposer has a ±30 V breakdown limit for each control channel. Exceeding this limitation is very likely to result in a short to ground in the trench capacitor for that channel. A current-voltage characteristic corresponding to a diode is indicative of a short between metal and silicon. A strictly ohmic short is usually caused by direct metal to metal shorts either on chip or in the wiring.

Figure 1 shows the electrode labels for the HOA-2 trap. In this figure the rf trace comes off in the bottom left position, which defines a specific orientation. This can also be determined from the direction indicated by the gold tab on the package.

In Table 2 the mapping between the electrodes and the package pins are listed. This should be combined with the user's particular vacuum wiring netlist to generate an end-to-end electrode-to-control channel netlist in order to accommodate the waveform files discussed in Section 5.

The axial secular frequency generated by a "unit set" of control voltages is:

$$\omega_{axial} = \sqrt{\frac{D\ U\ q}{m}},$$

where D depends on the particular unit voltage set (the geometry, electrodes used, etc.), U is the scale of the voltage applied, q is the charge of the ion, and m is the mass of the ion. This formula can be used to determine the effect of scaling the waveforms provided in Section 5. The secular frequencies in the radial directions will be impacted by the addition of the control voltages.

5 Control voltage solutions

5.1 Trap Simulations

Boundary element simulations are employed to calculate the charge on all trap electrodes. From the boundary element solutions, the electrostatic potential $U_i(\mathbf{x})$ at location $\mathbf{x}$ generated by an electrical potential of 1 V on electrode i while all other electrodes are at ground potential is calculated. Using the superposition principle, the potential generated by any set of voltages on the trap electrodes can be calculated from the set of U_i potentials. In addition, the

Package Pad	Package Grid	Interposer Signal	Electrode	Package Pad	Package Grid	Interposer Signal	Electrode	Package Pad	Package Grid	Interposer Signal	Electrode
94	B05	12	G01	35	M06	43	Q06	74	C12	86	Q39
33	N05	39	G02	9	F03	7	Q07	99	B03	22	Q40
10	F02	11	G03	18	J01	44	Q08	14	G01	26	rf
23	M01	40	G04	91	B06	6	Q09	82	B09	87	T1
83	A09	89	G05	36	N06	45	Q10	53	L12	65	T2
44	M09	62	G06	6	D01	5	Q11	4	C01	13	T3
73	B13	90	G07	21	K02	46	Q12	45	N10	64	T4
60	H12	61	G08	90	A06	4	Q13	67	F11	88	T5
12	G02	23	GND	37	M07	47	Q14	54	L13	63	T6
13	G03	25	GND	7	E02	3	Q15	78	B11	79	Y01
38	L07	49	GND	20	K01	48	Q16	65	F13	78	Y02
63	G11	75	GND	89	A07	2	Q17	77	A12	77	Y03
88	C07	99	GND	19	J02	50	Q18	95	A04	14	Y04
81	A10	85	L01	8	E01	1	Q19	96	B04	16	Y05
66	F12	84	L02	58	J13	51	Q20	3	C02	15	Y06
80	B10	83	L03	69	E12	100	Q21	97	A03	18	Y07
75	A13	82	L04	39	N07	52	Q22	11	F01	17	Y08
79	A11	81	L05	70	D13	98	Q23	29	N03	31	Y09
76	B12	80	L06	57	J12	53	Q24	25	N01	32	Y10
2	B01	19	L07	87	B07	97	Q25	28	M03	29	Y11
98	A02	20	L08	40	N08	54	Q26	26	M02	30	Y12
32	M05	37	L09	71	D12	96	Q27	15	H01	28	Y13
17	H03	38	L10	56	K13	55	Q28	27	N02	27	Y14
31	N04	35	L11	86	A08	95	Q29	100	A01	24	Y15
24	L02	36	L12	41	M08	56	Q30	46	M10	66	Y16
30	M04	33	L13	68	E13	94	Q31	62	G12	73	Y17
16	H02	34	L14	59	H11	57	Q32	49	M11	72	Y18
47	N11	68	L15	85	B08	93	Q33	50	N13	74	Y19
61	H13	67	L16	42	L08	58	Q34	51	M12	71	Y20
93	A05	10	Q01	72	C13	92	Q35	64	G13	76	Y21
34	L06	41	Q02	55	K12	59	Q36	48	N12	70	Y22
5	D02	9	Q03	84	C08	91	Q37	1	B02	21	Y23
22	L01	42	Q04	43	N09	60	Q38	52	M13	69	Y24
92	C06	8	Q05								

Table 2: Netlist ordered by trap electrode

Package Pad	Package Grid	Interposer Signal	Electrode	Package Pad	Package Grid	Interposer Signal	Electrode	Package Pad	Package Grid	Interposer Signal	Electrode
1	B02	21	Y23	35	M06	43	Q06	68	E13	94	Q31
2	B01	19	L07	36	N06	45	Q10	69	E12	100	Q21
3	C02	15	Y06	37	M07	47	Q14	70	D13	98	Q23
4	C01	13	T3	38	L07	49	GND	71	D12	96	Q27
5	D02	9	Q03	39	N07	52	Q22	72	C13	92	Q35
6	D01	5	Q11	40	N08	54	Q26	73	B13	90	G07
7	E02	3	Q15	41	M08	56	Q30	74	C12	86	Q39
8	E01	1	Q19	42	L08	58	Q34	75	A13	82	L04
9	F03	7	Q07	43	N09	60	Q38	76	B12	80	L06
10	F02	11	G03	44	M09	62	G06	77	A12	77	Y03
11	F01	17	Y08	45	N10	64	T4	78	B11	79	Y01
12	G02	23	GND	46	M10	66	Y16	79	A11	81	L05
13	G03	25	GND	47	N11	68	L15	80	B10	83	L03
14	G01	26	rf	48	N12	70	Y22	81	A10	85	L01
15	H01	28	Y13	49	M11	72	Y18	82	B09	87	T1
16	H02	34	L14	50	N13	74	Y19	83	A09	89	G05
17	H03	38	L10	51	M12	71	Y20	84	C08	91	Q37
18	J01	44	Q08	52	M13	69	Y24	85	B08	93	Q33
19	J02	50	Q18	53	L12	65	T2	86	A08	95	Q29
20	K01	48	Q16	54	L13	63	T6	87	B07	97	Q25
21	K02	46	Q12	55	K12	59	Q36	88	C07	99	GND
22	L01	42	Q04	56	K13	55	Q28	89	A07	2	Q17
23	M01	40	G04	57	J12	53	Q24	90	A06	4	Q13
24	L02	36	L12	58	J13	51	Q20	91	B06	6	Q09
25	N01	32	Y10	59	H11	57	Q32	92	C06	8	Q05
26	M02	30	Y12	60	H12	61	G08	93	A05	10	Q01
27	N02	27	Y14	61	H13	67	L16	94	B05	12	G01
28	M03	29	Y11	62	G12	73	Y17	95	A04	14	Y04
29	N03	31	Y09	63	G11	75	GND	96	B04	16	Y05
30	M04	33	L13	64	G13	76	Y21	97	A03	18	Y07
31	N04	35	L11	65	F13	78	Y02	98	A02	20	L08
32	M05	37	L09	66	F12	84	L02	99	B03	22	Q40
33	N05	39	G02	67	F11	88	T5	100	A01	24	Y15
34	L06	41	Q02								

Table 3: Netlist ordered by package pad

type	number	content
64 bit Integer	1	
64 bit Integer	1	N_{elec}: Number of electrodes
64 bit Integer	3	N_x, n_y, n_z: Number of samples along the x, y and z axis
64 bit Integer	1	Number of voltage sets (1)
64 bit double	3	re-mapped x-axis
64 bit double	3	re-mapped y-axis
64 bit double	3	stride along x-, y-, and z-axis
64 bit double	3	origin
64 bit Integer	2	
64 bit Integer	N_{elec}	Electrode mapping
64 bit double	$N_{\text{elec}} \cdot n_x \cdot n_y \cdot n_z$	Potential data

Table 4: File format of the binary voltage array files.

ponderomotive potential generated by the rf electrodes can be calculated using

$$U_{\text{pond}} = \frac{eV_{\text{rf}}^2}{4m\omega_{\text{rf}}} \nabla U_{\text{rf}} \cdot \nabla U_{\text{rf}},$$

where e is the electron charge, V_{rf} the voltage amplitude of the radio-frequency drive, m is the ion mass and U_{rf} is the electrostatic potential generated by the radiofrequency electrode.

5.1.1 Voltage Arrays

The values of $U_i(\mathbf{x})$ are calculated near the rf nodal lines for all the electrodes and saved in a file. We call these data Voltage Arrays.

File format The Voltage Arrays are saved in binary format and usually have a .bin file extension. A file header giving information about the number of electrodes, boundaries, axes is followed by an array of potential values. The full file format is specified in Table 4.

Electrode assignment The voltage array files contain the potential for the electrodes in the order given in Table 5.

5.2 Trapping solutions

Examples of trapping solutions and the methods to generate these can be found in the Mathematica file `VoltageGeneration.nb` . Here, we will describe general properties and strategies to generate trapping solutions.

Index	Electrode
1	Ground
2	rf
3 . . . 10	G01 . . . G08
11 . . . 26	L01 . . . L16
27 . . . 66	Q01 . . . Q40
67 . . . 72	T01 . . . T06
73 . . . 96	Y01 . . . Y24

Table 5: One-based index of the electrodes of the HOA-2 trap in voltage array files.

5.2.1 Principal Axes rotation

The principal axes of the trap can be rotated along the central linear section by applying opposite voltages to the outer control voltage rails Q39 and Q40 and adjusting the inner control electrodes to cancel the electric field along the rf nodal lines. Along the central linear section a static solution can be found that rotates the principal axes and cancels the field along the nodal lines.

In the loading zones, rotation of the principal axes can be achieved by applying opposing voltages to electrodes L07, L08, and L15, L16. However, it is not possible to find a single static solution that rotates the principal axes everywhere in the trap and achieves a vanishing electrical field along all nodal lines. However, this is not necessary for the operation of the trap. Instead, when creating a shuttling solution, the residual electrical field generated by global rotation solutions can be compensated for the trapping locations.

5.2.2 Trapping solutions in the center of slotted region

The simplest way to generate a trapping solution in the center of the linear section of the trap is to apply $-1\,\mathrm{V}$ to one pair of inner control electrodes (Q19 and Q20) and choose the voltage on the outer electrodes (Q39 and Q40) to cancel the electric field at the trapping location ($V_{Q19} = V_{Q20} = -1\,\mathrm{V}$, $V_{Q39} = -0.914\,\mathrm{V}$, $V_{Q40} = -0.910\,\mathrm{V}$). For ytterbium, this generates an axial trap with a trap frequency of 500 kHz and trap depth of 60 meV. The trap frequency and depth can now be scaled by scaling all voltages.

The trap frequency and depth can be increased without increasing the voltage budget by assigning a positive voltage to the neighboring inner control voltages. For example ($V_{Q19} = V_{Q20} = -1\,\mathrm{V}$, $V_{Q09} = \cdots = V_{Q18} = V_{Q21} = \cdots = V_{Q30} = 1\,\mathrm{V}$, $V_{Q39} = -0.445\,\mathrm{V}$, $V_{Q40} = -0.442\,\mathrm{V}$) generates a trap frequency of 700 kHz with a depth of about 120 meV for ytterbium.

The trap depth can be increased without increasing the trap frequency by assigning a negative potential to multiple electrode pairs.

Compensation solutions to apply an electric field in y and z directions uniform along the nodal line can be achieved by applying (or adding) antisymmetric or symmetric voltages to the outer control electrodes, respectively. For exam-

ple $V_{Q39} = -0.777$, $V_{Q40} = -0.775$ will generate a vertical electrical field of $1000\,\mathrm{V/m}$. Please note that because the outer electrodes generate a quadrupole field with nodal line *above* the rf nodal line, the electric field at the rf nodal line will point towards the surface for a positive voltage applied to the outer electrodes. The difference between V_{Q39} and V_{Q40} could either be due to the asymmetry of the trap — the rf feed breaks the overall symmetry — or due to asymmetry of the chosen meshing for in the boundary element simulation. To pull an ion closer to the surface, a negative voltage has to be applied to the outer electrodes.

Voltages $V_{Q39} = -0.987$, $V_{Q40} = 0.985$ will create a lateral electrical field of strength $1000\,\mathrm{V/m}$ along the positive y-axis.

5.2.3 Shuttling solutions through the trap

Shuttling solutions are generated using the global minimization of a cost function for each trap location. We define the overall cost of a voltage solution $\mathbf{v}$ as

$$\begin{aligned} C(\mathbf{v}) = \; & w_f(f_{\mathrm{trap}}(\mathbf{v}) - f_{\mathrm{target}})^2 \\ & + w_E(E_x(\mathbf{v})^2 + E_y(\mathbf{v})^2 + E_z(\mathbf{v})^2) \\ & + w_{\mathrm{a}} A(\mathbf{v}) \\ & + w_{\mathrm{v}}(\mathbf{v} - \mathbf{v}_{\mathrm{target}}) \cdot \mathbf{P}_{\mathrm{elec}}, \end{aligned}$$

where w_f, w_E, w_a and w_v are the individual weights for the trap frequency, residual electric fields, solution asymmetry, and electrode cost, respectively. $f_{\mathrm{trap}}(\mathbf{v})$ is the axial trap frequency generated by the voltage set and f_{target} the target trap frequency. $E_x(\mathbf{v})$, $E_y(\mathbf{v})$ and $E_z(\mathbf{v})$ are the residual electric fields at the trapping location in $x-$, $y-$ and $z-$direction, respectively. $A(\mathbf{v})$ is the asymmetry of the voltage set. This is the sum of the differences of the voltages on all electrode pairs. $\mathbf{v}_{\mathrm{target}}$ is a target voltage and $\mathbf{P}_{\mathrm{elec}}$ are the electrode penalties for the trapping location. Electrode penalties are intended to confine trapping solutions to those electrodes that generate the strongest fields at the trapping location. The electrode cost for electrode i at trapping location $\mathbf{x}$ is defined as the inverse of the absolute value of the electric field generated by this electrode at the trapping location:

$$P_{\mathrm{elec},i} = \frac{1}{(\nabla U_i(\mathbf{x}))^2}.$$

The asymmetry cost is included in the cost function to prevent a degeneracy of the cost in the parameter space and ensure that a unique global cost minimum exists. Therefore, the weights for the essential properties of a trapping solution w_f, w_E are chosen to be much larger than the weights for asymmetry and electrode costs.

Shuttling voltage sets are then created by minimizing this weight function for an equidistant sequence of points along the rf nodal line.

The outer control electrodes are held at a constant voltage when calculating shuttling solutions and due to the electrode cost in the cost function only electrodes close to the trapping location are used. Thus it is possible to combine shuttling solutions for different locations to shuttle multiple ions trough the trap simultaneously.

5.2.4 Shuttling solutions with principal axes rotation

Ideally, one would be able to add a global rotation solution that rotates the principal axes everywhere in the trap and does not have residual electrical fields on the rf nodal lines to the shuttling solution. However, due to the limited number of degrees of freedom, such a global rotation solution does not exist. Instead, shuttling solutions with principal axes rotation are generated by adding the global rotation solution described in Section 5.2.1 to the target voltage solution. The optimization process will then minimize the residual electrical fields at each trapping location and thus compensate for the residual electrical field of the global rotation solution.

6 Specifications and absolute maximum ratings

Rf Electrode	
Maximal voltage on rf electrodes	$\pm 300\,\mathrm{V_{pk}}$
Capacitance of trap chip	$\approx 7\,\mathrm{pF}$
Capacitance of assembly	$\approx 10\,\mathrm{pF}$
DC Control electrodes	
Maximal voltage on dc electrodes	$\pm 20\,\mathrm{V}$
Maximal leakage current through trench capacitor at $\pm 10\,\mathrm{V}$	5nA (non-ohmic)
Number of control voltages needed	94
Trench capacitors (all dc control electrodes)	
Capacitance	$1.05\,\mathrm{nF}$
Series inductance	$\approx 50\,\mathrm{pH}$
Series resistance ($20\,^\circ\mathrm{C}$)	$4\,\Omega$
Maximum voltage	$\pm 20\,\mathrm{V}$
Breakdown voltage	$> \pm 30\,\mathrm{V}$
Mechanical	
Trap surface to CPGA seal ring vertical distance (Limited by CPGA tolerances)	1.6 mm to 1.9 mm
Optical	
Numerical aperture skimming the surface perpendicular to trap axis	0.11
Numerical aperture skimming the surface at 45° to trap axis	0.08
Numerical aperture vertically through central slot	0.25

7 Attachments

The files listed below supply additional information about the trap, trapping potentials and voltage solution generation.

`CPG10039.pdf`	CPGA carrier datasheet
`H21D.pdf`	datasheet of die attachment epoxy
`packageStandard_v2.0.pdf`	MQCO packaging standard
`HOA2-RS1096.pdf`	Trap schematic. Drawing is realistic and to scale. 1 inch $\widehat{=}$10 mm. Measurements can be done using Adobe Reader's Measuring Tool.
`HOA2.accdb`	HOA-2 netlist as Microsoft Access database
`HOA2Netlist by electrode.txt`	HOA-2 netlist as textfile sorted by electrode name.
`HOA2Netlist by package.txt`	HOA-2 netlist as textfile sorted by package pad.
`HOA2Netlist.xslx`	HOA-2 netlist as Microsoft Excel file.
VoltageArray/	
`RS1096_12400_1.bin`	Voltage Array along linear section. $-2308.5\,\mu\text{m} \le x \le 2308.5\,\mu\text{m}$, $-30\,\mu\text{m} \le y \le 30\,\mu\text{m}$, $40\,\mu\text{m} \le z \le 110\,\mu\text{m}$
`RS1096_12400_L1.bin`	Voltage Array of loading leg close to pad 1. $0\,\mu\text{m} \le x \le 560\,\mu\text{m}$, $-30\,\mu\text{m} \le y \le 30\,\mu\text{m}$, $40\,\mu\text{m} \le z \le 110\,\mu\text{m}$. x-axis is aligned with the rf nodal line. $x = 0\,\mu\text{m}$ is at the junction center.
`RS1096_12400_L2.bin`	Voltage Array of loading leg close to pad 25.
`RS1096_12400_L3.bin`	Voltage Array of loading leg close to pad 50.
`RS1096_12400_L4.bin`	Voltage Array of loading leg close to pad 75.
`TrapEvaluation.m`	Mathematica package containing functions for loading of Voltage Arrays and evaluation of potentials.
`HOA2-VoltageGeneration.nb`	Mathematica file demonstrating the procedures to visualize trap potentials and calculate trapping solutions.
VoltageSolutions/	
`ShuttleR1.txt`	Shuttling solution from left junction center to right junction center *without* principal axes rotation. (one solution every 5 μm)
`ShuttleR1.cdf`	Mathematica animation file showing axial potential generated by `ShuttleR1.txt`

`ShuttleL1.txt`	Shuttling solution between trap center and loading hole near package pad 1 *without* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL1.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL1.txt`
`ShuttleL2.txt`	Shuttling solution between trap center and loading hole near package pad 25 *without* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL2.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL2.txt`
`ShuttleL3.txt`	Shuttling solution between trap center and loading hole near package pad 50 *without* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL3.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL3.txt`
`ShuttleL4.txt`	Shuttling solution between trap center and loading hole near package pad 75 *without* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL4.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL4.txt`
`ShuttleR1Rot.txt`	Shuttling solution from left junction center to right junction center *without* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleR1Rot.cdf`	Mathematica animation file showing axial potential generated by `ShuttleR1Rot.txt`
`ShuttleL1Rot.txt`	Shuttling solution between trap center and loading hole near package pad 1 *with* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL1Rot.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL1Rot.txt`
`ShuttleL2Rot.txt`	Shuttling solution between trap center and loading hole near package pad 25 *with* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL2Rot.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL2Rot.txt`
`ShuttleL3Rot.txt`	Shuttling solution between trap center and loading hole near package pad 50 *with* principal axes rotation. (one solution every $5\,\mu$m)

`ShuttleL3Rot.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL3Rot.txt`
`ShuttleL4Rot.txt`	Shuttling solution between trap center and loading hole near package pad 75 *with* principal axes rotation. (one solution every $5\,\mu$m)
`ShuttleL4Rot.cdf`	Mathematica animation file showing axial potential generated by `ShuttleL4Rot.txt`

References

[1] D L Moehring, C Highstrete, D Stick, K M Fortier, R Haltli, C Tigges, and M G Blain. Design, fabrication and experimental demonstration of junction surface ion traps. *New Journal of Physics*, 13(7):075018, July 2011.

[2] G. Shu, G. Vittorini, A. Buikema, C. S. Nichols, C. Volin, D. Stick, and Kenneth R. Brown. Heating rates and ion-motion control in a Y-junction surface-electrode trap. *Physical Review A*, 89(6):062308, June 2014.

[3] G. Chen. On the Physics of Purple-Plague Formation, and the Observation of Purple Plague in Ultrasonically-Joined Gold-Aluminum Bonds. *IEEE Transactions on Parts, Materials and Packaging*, 3(4):149–153, December 1967.

HOA-2 Ion Trap Simulations

Initialization

First, we load a library with functions for loading of voltage arrays and potential evaluations, set the directory, and define a few constants

```
rootdir = SetDirectory[NotebookDirectory[]]
```

```
C:\Users\plmaunz\Documents\Trap-Simulations\HOA-2-delivery
```

```
Get["TrapEvaluation.m"]
```

Parameters

Voltages

RF Voltage

```
vRF = 250;
```

Constants

Fundamental Constants: electric charge, mass of $^{171}Yb^{+}$ in kg

```
qe = electronCharge;
IonAmu = 171;
mIon = IonAmu amu;
```

Trap parameters: RF frequency, and ion--electrode spacing for RF

```
ωRF = 2 π * (45.0 * 10^6);
```

This is a ponderomotive constant. The potential energy is then $\psi_p = \text{pPondConst} \times E^2$. Note that the electric field from CPO is in V/mm, and the units of $\frac{q^2 V^2}{4 m_{\text{ion}} \omega_{\text{RF}}^2}$ is $J \times m^2$. The factor of 10^6 is to convert the units to $J \times \text{mm}^2$.

```
pPondConst = vRF^2 qe^2 / (4 mIon ωRF^2) * 10^6;
```

Default Plot Styles

Import Simulation Results

Import the voltage array data. There are independent voltage array files for rectangualar regions along the main linear section and along the 4 loading arms.
The coordinate system in each of the five regions has the rf nodal line aligned along the x-axis, the y-axis is parallel to the trap surface and the z-axis is perpendicular to the trap surface.
The Origins define the coordinate of the first point of the voltage array in this coordinate system. The coordinate systems are different for the 5 regions.

```
Filenames = {"RS1096_12400_1", "RS1096_12400_L1",
  "RS1096_12400_L2", "RS1096_12400_L3", "RS1096_12400_L4"}
Origins = {{-2.350, -0.03, 0.04}, {-0.1, -0.03, 0.04},
   {-0.1, -0.03, 0.04}, {-0.1, -0.03, 0.04}, {-0.1, -0.03, 0.04}};
```

```
{RS1096_12400_1, RS1096_12400_L1,
 RS1096_12400_L2, RS1096_12400_L3, RS1096_12400_L4}
```

Reading and interpolating the data takes a considerable amout of time. Therefore, we do this once and write the interpolation functions to a file with the filename of the voltage array and the extension ".mx". These files are not compatible across different versions of *Mathematica*. Should there be problems, please delete the ".mx" files. They will be regenerated once the command is run again.

```
Clear[HeaderList, InterpolantsList, ROIList, CenterList]
LoadInterpolantsList[Filenames, HeaderList,
  InterpolantsList, ROIList, CenterList, 64, Origins];
```

```
Reading cached data.

Reading cached data.

Reading cached data.

Reading cached data.

Reading cached data.
```

```
Header = HeaderList[[1]];
Header["electrodes"]
```

```
96
```

Calculate ROI and expected trap location

Minimal and maximal coordinate values for the different regions, that is interval in which voltage data is available. Outside this region extrapolation will be used which will NOT lead to acceptable results.
The region enumerates the 5 regions of the the trap [main linear region, loading arm near pad 1 (top left), loading arm near pad 25 (bottom left), loading arm near pad 50 (bottom right), loading arm near pad 75 (top right)]

```
xmin[region_] = Indexed[{-2.3085, 0., 0., 0., 0.}, region];
xmax[region_] = Indexed[{2.3085, 0.560, 0.560, 0.560, 0.560}, region];
ymin[region_] = -0.03;
ymax[region_] = 0.03;
zmin[region_] = 0.04;
zmax[region_] = 0.11;
```

RF Ponderomotive Potential

3-Dimensional

The simulation contains the rf electrode as electrode 2. We have to take the dotproduct of the gradient and multiply with the ponderomotive constant.
The standard version uses the fixed rf voltage defined in the Initialization section, the second version allows one to specify a rf voltage

```
potentialrfInternal[x_, y_, z_] = Table[With[{rf = InterpolantsList[[i, 2]]},
    With[{vec = {D[rf[x, y, z], x], D[rf[x, y, z], y], D[rf[x, y, z], z]}},
     pPondConst * 1/qe * vec.vec]], {i, 1, 5}];
potentialrf[x_, y_, z_, region_] := potentialrfInternal[x, y, z][[region]]
potentialrf[x_, y_, z_, region_, myvRF_] :=
 potentialrfInternal[x, y, z][[region]] (myvRF/vRF)^2
```

Plot the ponderomotive potential in the plane perpendicular to the nodal line.

```
With[{r = 1},
 Animate[ContourPlot[Evaluate[potentialrf[x, y, z, r]], {y, ymin[r], ymax[r]},
   {z, zmin[r], zmax[r]}, ImageSize → 500], {x, xmin[r], xmax[r]}]]
```

Find the position of the nodal point in this plane. The position in y-direction is expected close to zero, however, the asymmetry of the trap and simulation leads to a nodal position at a different values. To ensure that we always look at the potential at the node, we find the nodal position in y and z directions.

```
rfNodeYZ[myx_, region_] :=
 NMinimize[{potentialrf[myx, y, z, region], ymin[region] < y < ymax[region],
    zmin[region] < z < zmax[region]}, {y, z}][[2]]
```

We look for the nodal position every 5 μm, and will use an Interpolating function. Again this calculation is expensive and will be cached in the file “nodetable.mx”. If a recalculation is necessary or problems appear, please delete this file.

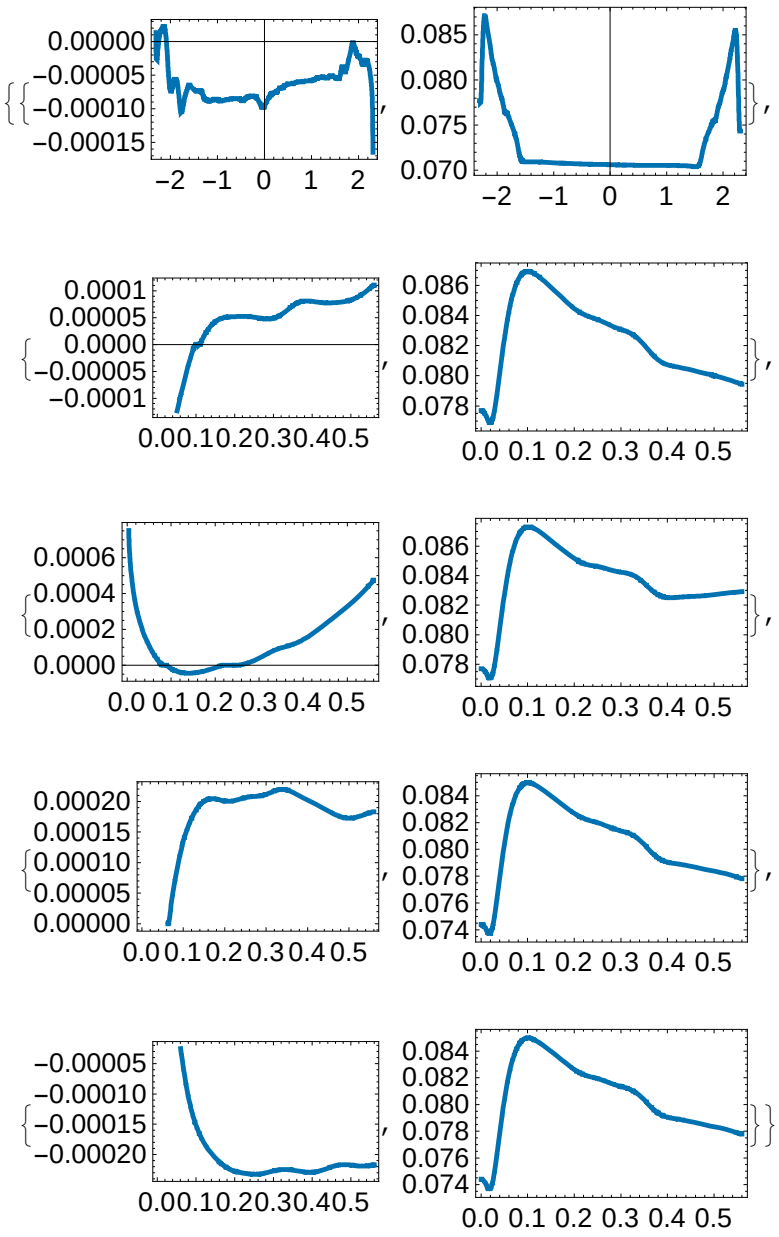
```
If[FileExistsQ["nodetable.mx"],
 Get["nodetable.mx"],
 nodetable =
  Table[Table[{x, y, z} /. rfNodeYZ[x, r], {x, xmin[r], xmax[r], 0.005}], {r, 1, 5}];
 DumpSave["nodetable.mx", nodetable]]
```

nodeY[xpos, region] and nodeZ[xpos, region] give the nodal position as a function of the position along the x-axis for the different regions of the trap.

```
nodeYInterpol = Table[Interpolation[nodetable[[r, All, {1, 2}]]], {r, 1, 5}];
nodeZInterpol = Table[Interpolation[nodetable[[r, All, {1, 3}]]], {r, 1, 5}];
nodeY[x_, region_] := nodeYInterpol[[region]][x]
nodeZ[x_, region_] := nodeZInterpol[[region]][x]
```

Position of the node in y and z direction for the 5 regions of the trap.

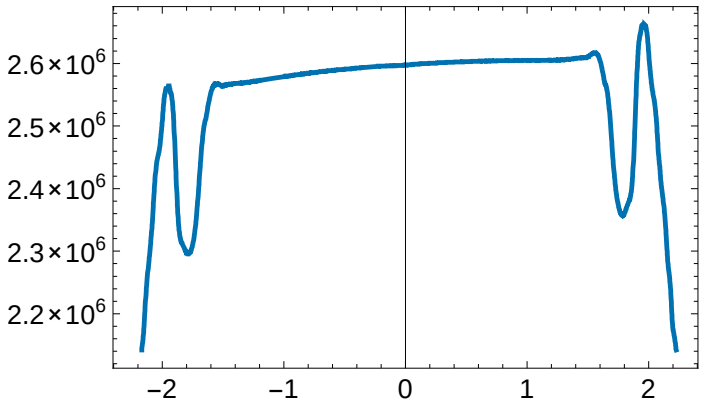
```
Table[{Plot[nodeY[x, r], {x, xmin[r], xmax[r]}],
  Plot[nodeZ[x, r], {x, xmin[r], xmax[r]}]}, {r, 1, 5}]
```



We calculate the higher radial trap frequency along the nodal line

```
MaxTrapFreqTable =
  Table[Table[{myx, Max[Re[CharacterizePotential[potentialrf[x, y, z, r],
         {myx, nodeY[myx, r], nodeZ[myx, r]}, IonAmu]][[1]]]]},
    {myx, xmin[r], xmax[r], 0.005}], {r, 1, 5}];
MaxTrapFreqInterpol = Table[Interpolation[MaxTrapFreqTable[[r]]], {r, 1, 5}];
MaxTrapFreq[x_, region_] := MaxTrapFreqInterpol[[region]][x]
```

```
With[{r = 1}, Plot[MaxTrapFreq[x, r], {x, xmin[r], xmax[r]}]]
```



General DC Potential

DCPotential at point x, y , z for voltage solution voltages in region region. Voltages is a vector of 96 voltages in volt for the 96 electrodes.
Electrode 1: GND
2: RF
3-10: G01-G08
11-26: L01-L16
27-66: Q01-Q40
67-72: T01-T06
73-96: Y01-Y24

```
DCPotential[x_, y_, z_, voltages_, region_] :=
 Sum[InterpolantsList[[region, i]][x, y, z] * voltages[[i]],
  {i, 1, Length[InterpolantsList[[region]]]}]
```

Global Rotation Solution

Here we calculate a global solution which rotates the principal axes of the trap and tries to minimize the field anywhere on the rf nodal line simultaneously. Because a solution that has vanishing field anywhere along the nodal line does not exist, this is just an approximation. When a solution is calculated for a specific position, thes voltages have to be included for the fields to be compensated at the position of the node.
The electrode mask eMaskRotation defines which electrode voltages are to be adjusted. The voltages on the outer electrodes are fixed at 1V and -1V.

```
eMaskRotation = Flatten[{ConstantArray[0, 2], Array[s, 14], 1, -1,
    Array[s, 6, 15], 1, -1, Array[s, 38, 21], 1, -1, Array[s, 30, 59]}];
```

The voltage mask is just a vector of all degrees of freedom

```
vMaskRotation = Array[s, 88];
```

The Weight function for the global minimzation includes values with different physical unit. Therefore we have to use weights for the different quantities to be able to compare them. In addition we perhaps care more about fields in y- and z-direction than in x-direction.

```
{xWeight, yWeight, zWeight} = {1, 10, 10}
```

```
{1, 10, 10}
```

The solution is calculated using a global minimzation algorithm. The Weight function includes terms that favor vanishing field anywhere on the nodal lines.

```
WeightFunctionRotation[v_] := Sum[With[{pot = DCPotential[x, y, z, v, r]},
    (xWeight D[pot, x]^2 + yWeight D[pot, y]^2 + zWeight D[pot, z]^2) /.
     {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}],
  {r, 1, 5}, {posx, xmin[r], xmax[r], 0.01}]
```

This is the rotation solution

```
rotSolution = NMinimize[WeightFunctionRotation[eMaskRotation], vMaskRotation][[2]]
```

```
{s[1] → -0.881445, s[2] → 0.876697, s[3] → -0.891672, s[4] → 0.886016,
 s[5] → -0.883839, s[6] → 0.878905, s[7] → -0.89514, s[8] → 0.889379,
 s[9] → -0.74144, s[10] → 0.797616, s[11] → -0.743762, s[12] → 0.783921,
 s[13] → -0.732757, s[14] → 0.796795, s[15] → -0.891723, s[16] → 0.81929,
 s[17] → -0.814198, s[18] → 0.765873, s[19] → -0.827604, s[20] → 0.756512,
 s[21] → -0.89314, s[22] → 0.884578, s[23] → -0.887994, s[24] → 0.882095,
 s[25] → -0.890681, s[26] → 0.882858, s[27] → -0.888671, s[28] → 0.881562,
 s[29] → -0.8886, s[30] → 0.881841, s[31] → -0.887964, s[32] → 0.880886,
 s[33] → -0.88722, s[34] → 0.880982, s[35] → -0.887571, s[36] → 0.879529,
 s[37] → -0.88694, s[38] → 0.880017, s[39] → -0.887231, s[40] → 0.879947,
 s[41] → -0.886007, s[42] → 0.879334, s[43] → -0.884408, s[44] → 0.880381,
 s[45] → -0.884893, s[46] → 0.879339, s[47] → -0.884686, s[48] → 0.880057,
 s[49] → -0.88334, s[50] → 0.878964, s[51] → -0.885454, s[52] → 0.880699,
 s[53] → -0.880598, s[54] → 0.877189, s[55] → -0.888549, s[56] → 0.883318,
 s[57] → -0.875101, s[58] → 0.872556, s[59] → -0.888373, s[60] → 0.887244,
 s[61] → -0.941015, s[62] → 0.935467, s[63] → -0.874395, s[64] → 0.870839,
 s[65] → -0.745576, s[66] → 0.827439, s[67] → -0.687197, s[68] → 0.802298,
 s[69] → -0.551194, s[70] → 0.645185, s[71] → -0.173717, s[72] → 0.304004,
 s[73] → -0.844458, s[74] → 0.759104, s[75] → -0.81444, s[76] → 0.696278,
 s[77] → -0.649415, s[78] → 0.554601, s[79] → -0.304851, s[80] → 0.173121,
 s[81] → -0.531349, s[82] → 0.529017, s[83] → -0.857878, s[84] → 0.857186,
 s[85] → -0.854049, s[86] → 0.852079, s[87] → -0.814364, s[88] → 0.811495}
```

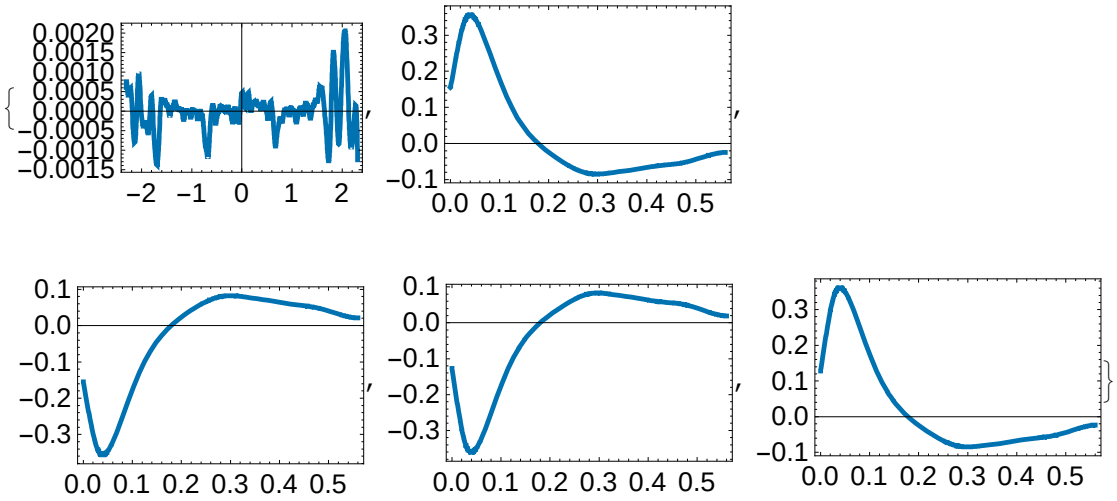
And these are the voltages

```
rotationVoltages = eMaskRotation /. rotSolution
```

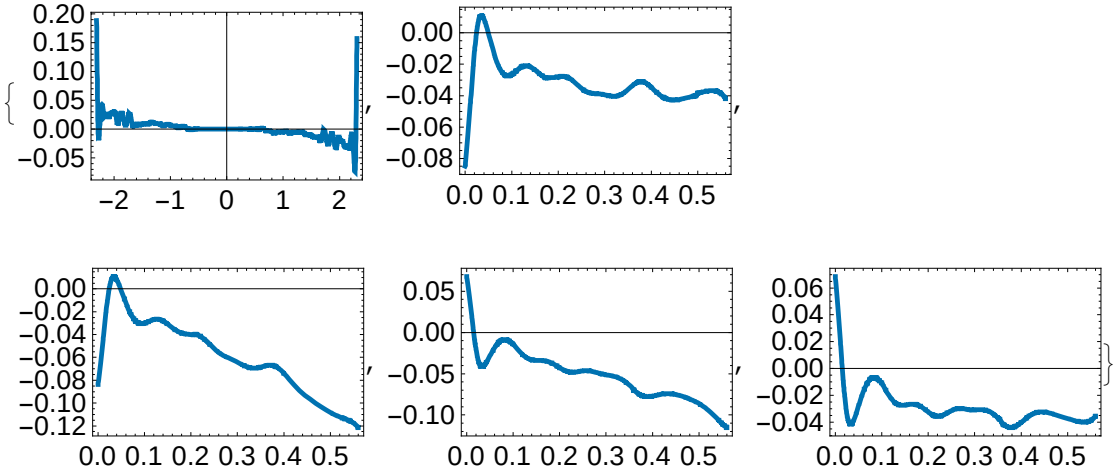
```
{0, 0, -0.881445, 0.876697, -0.891672, 0.886016, -0.883839, 0.878905, -0.89514,
 0.889379, -0.74144, 0.797616, -0.743762, 0.783921, -0.732757, 0.796795,
 1, -1, -0.891723, 0.81929, -0.814198, 0.765873, -0.827604, 0.756512,
 1, -1, -0.89314, 0.884578, -0.887994, 0.882095, -0.890681, 0.882858,
 -0.888671, 0.881562, -0.8886, 0.881841, -0.887964, 0.880886, -0.88722,
 0.880982, -0.887571, 0.879529, -0.88694, 0.880017, -0.887231, 0.879947,
 -0.886007, 0.879334, -0.884408, 0.880381, -0.884893, 0.879339, -0.884686,
 0.880057, -0.88334, 0.878964, -0.885454, 0.880699, -0.880598, 0.877189,
 -0.888549, 0.883318, -0.875101, 0.872556, 1, -1, -0.888373, 0.887244,
 -0.941015, 0.935467, -0.874395, 0.870839, -0.745576, 0.827439, -0.687197,
 0.802298, -0.551194, 0.645185, -0.173717, 0.304004, -0.844458, 0.759104,
 -0.81444, 0.696278, -0.649415, 0.554601, -0.304851, 0.173121, -0.531349,
 0.529017, -0.857878, 0.857186, -0.854049, 0.852079, -0.814364, 0.811495}
```

Let's plot the remaining fields in V/mm for the 5 different regions and the x-, y- and z-direction. As we see there are considerable residual fields in the junction abd loading regions. Thus a trapping solution using this rotation solutions has to take these into account.

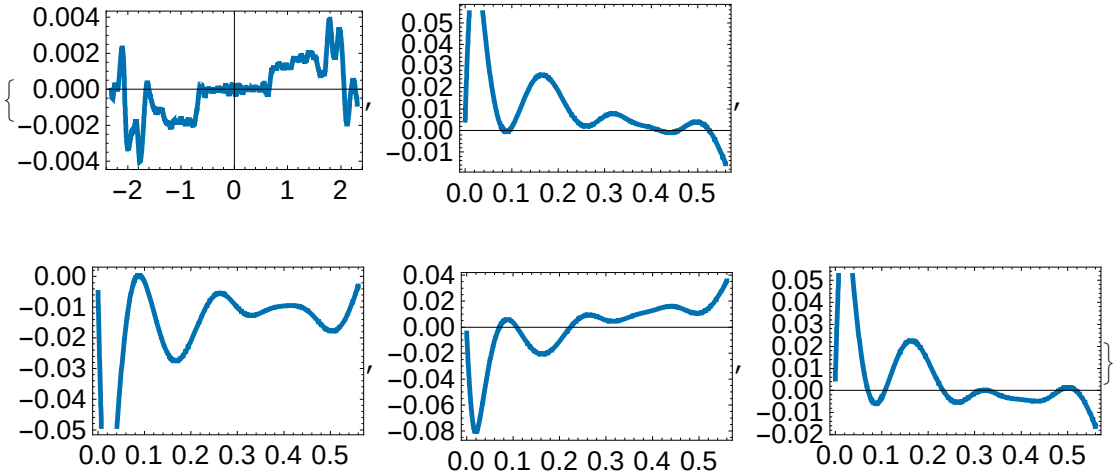
```
Table[Plot[Evaluate[With[{pot = DCPotential[x, y, z, rotationVoltages, r]},
    D[pot, x] /. {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}]],
  {posx, xmin[r], xmax[r]}, PlotRange → All], {r, 1, 5}]
```



```
Table[Plot[Evaluate[With[{pot = DCPotential[x, y, z, rotationVoltages, r]},
    D[pot, y] /. {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}]],
  {posx, xmin[r], xmax[r]}, PlotRange → All], {r, 1, 5}]
```



```
Table[Plot[Evaluate[With[{pot = DCPotential[x, y, z, rotationVoltages, r]},
    D[pot, z] /. {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}]],
  {posx, xmin[r], xmax[r]}], {r, 1, 5}]
```



```
(* With[{r=1},Table[ContourPlot[
   Evaluate[potentialrf[x,y,z,r]+DCPotential[x,y,z,rotationVoltages,r]],
   {y,ymin[r],ymax[r]},{z,zmin[r],zmax[r]},
   ImageSize→500],{x,xmin[r],xmax[r],0.07}]]*)
```

Dedicated Solutions in the central Quantum region

Prerequisites

General definitions for shuttling. ElectrodeNames give the names of the electrodes in the same order as the voltage solution and is used for printing the solutions in a file.

```
ElectrodeNames = Flatten[{Table["G" <> IntegerString[i, 10, 2], {i, 1, 8}],
   Table["L" <> IntegerString[i, 10, 2], {i, 1, 16}],
   Table["Q" <> IntegerString[i, 10, 2], {i, 1, 40}],
   Table["T" <> IntegerString[i, 10, 1], {i, 1, 6}],
   Table["Y" <> IntegerString[i, 10, 2], {i, 1, 24}]}]
```

```
{G01, G02, G03, G04, G05, G06, G07, G08, L01, L02, L03, L04, L05, L06, L07, L08,
 L09, L10, L11, L12, L13, L14, L15, L16, Q01, Q02, Q03, Q04, Q05, Q06, Q07, Q08,
 Q09, Q10, Q11, Q12, Q13, Q14, Q15, Q16, Q17, Q18, Q19, Q20, Q21, Q22, Q23,
 Q24, Q25, Q26, Q27, Q28, Q29, Q30, Q31, Q32, Q33, Q34, Q35, Q36, Q37, Q38,
 Q39, Q40, T1, T2, T3, T4, T5, T6, Y01, Y02, Y03, Y04, Y05, Y06, Y07, Y08, Y09,
 Y10, Y11, Y12, Y13, Y14, Y15, Y16, Y17, Y18, Y19, Y20, Y21, Y22, Y23, Y24}
```

```
AllElectrodeNames = Flatten[{"GND", "RF", ElectrodeNames}];
```

globalProcess variable is used to show progress on the sometimes slow calculations. There is only one global variable and all Progress bars will move no matter which of the expressions that support the progress bar are used.

```
globalProgress = 0;
```

This function plots the axial trapping potential for each solution in sol.

```
solutionFrame[sol_, eMask_, region_] :=
 With[{bundle = Evaluate[DCPotential[myx, nodeY[myx, region],
       nodeZ[myx, region], eMask /. sol[[2]], region]]},
  With[{minpos = 0}, With[{minvalue = DCPotential[minpos, nodeY[minpos, region],
       nodeZ[minpos, region], eMask /. sol[[2]], region]},
    Plot[Evaluate[bundle - minvalue], {myx, xmin[region], xmax[region]},
     PlotRange → {-0.01, 0.3}]]]]
```

Write the solution to a file.

```
writeSolution[filename_, sol_, eMask_, region_] := Module[{stream},
  stream = OpenWrite[filename];
  Export[stream, {ElectrodeNames}, "Table"];
  WriteString[stream, "\n"];
  Export[stream, Table[
    eMask[sol[[i, 1]]][[3 ;; 96]] /. sol[[i, 3]], {i, 1, Length[sol]}], "Table"];
  Close[stream];]
```

Weights for the different contributions to the WeightFunction

```
{freqWeight, fieldWeight, symWeight, eWeight} = {100, 1, 10^-2, 10^-5}
```

$$\left\{100,\ 1,\ \frac{1}{100},\ \frac{1}{100\,000}\right\}$$

Calculate the length of the field vector generated by each electrode at the node at position posx along the x - axis. The inverse of this quantity is then used as an additional weight to have the solution favor those electrodes that generate the strongest field at the given position. Or, in other words, prevent high voltages at far away electrodes.

```
electrodepenalty[posx_, region_] :=
 Table[With[{elec = DCPotential[x, y, z, UnitVector[96, e], region]},
   1 / Sqrt[(D[elec, z])^2 + (D[elec, y])^2 + (D[elec, x])^2 /.
      {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}]], {e, 1, 96}]
```

A weight for asymmetry of a solution to favor symmetric solutions. In case of rotating solutions this function is used for the deviation of the rotating solution.

```
asymmetry[v_] := Sum[(v[[i]] - v[[i + 1]])^2, {i, 3, 95, 2}]
```

The WeightFunction for the global optimization for a trap at axial position posx [mm] with trap frequency freq [MHz], targetvoltages targetvoltage [V] for a given region of the trap. Contributions are:
freqWeight: Favoring the requested axial trap frequency
fieldWeight: Favoring vanishing DC fields at the nodal location
symWeight: Favoring symmetric solutions
eWeight: Favoring voltages on nearby electrodes

```
WeightFunction[v_, numPairs_, freq_, targetvoltage_, region_] := With[{posx = 0}, Abs[
   (freqWeight (CharacterizeAxialFrequency[DCPotential[x, y, z, v, region], {posx,
            nodeY[posx, region], nodeZ[posx, region]}, IonAmu] / 10^6 - freq)^2 +
     fieldWeight ((D[DCPotential[x, y, z, v, region], z])^2 +
        (D[DCPotential[x, y, z, v, region], y])^2 +
        (D[DCPotential[x, y, z, v, region], x])^2) + symWeight asymmetry[v] +
     eWeight Total[((v - targetvoltage[numPairs]) (v - targetvoltage[numPairs])).
        electrodepenalty[posx, region]]) /.
    {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}]]
```

Generate plots fields in V/m and frequency

```
verificationPlots[sol_, eMask_, region_] :=
 {ListPlot[Table[(globalProgress = i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], x] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "x-Field"],
  ListPlot[Table[(globalProgress = 1 / 4 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], y] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "y-Field"],
  ListPlot[Table[(globalProgress = 1 / 2 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], z] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "z-Field"],
  ListPlot[Table[(globalProgress = 3 / 4 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]}, {posx, CharacterizeAxialFrequency[DCPotential[x, y,
          z, eMask[posx] /. sol[[i, 3]], region], {posx, nodeY[posx, region],
          nodeZ[posx, region]}, IonAmu] / 10^6}]), {i, 1, Length[sol]}],
   AxesLabel → {"trap freq [MHz]", "pos [mm]"}, PlotLabel → "trap frequency"]}
```

Trapping Solutions zero outside target

Define target voltages for position along the nodal lines. This can be used to suggest trapping solutions that have a deeper trap depth or other intended properties.

```
targetvoltageCenter[numPairs_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]],
    ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}];
```

The electrodeMask defines the electrodes for which the voltage is to be adjusted to generate the trap.

```
eMaskCenter[numPairs_] := Flatten[{ConstantArray[0, 16], 0, 0, ConstantArray[0, 6],
    0, 0, ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]], Array[s, 2 numPairs, 2],
    ConstantArray[{0, 0}, 9 - Floor[numPairs / 2]], s[1], s[1], ConstantArray[0, 30]}];
```

Here we just define the number of degrees of freedom to match those in electrodeMask.

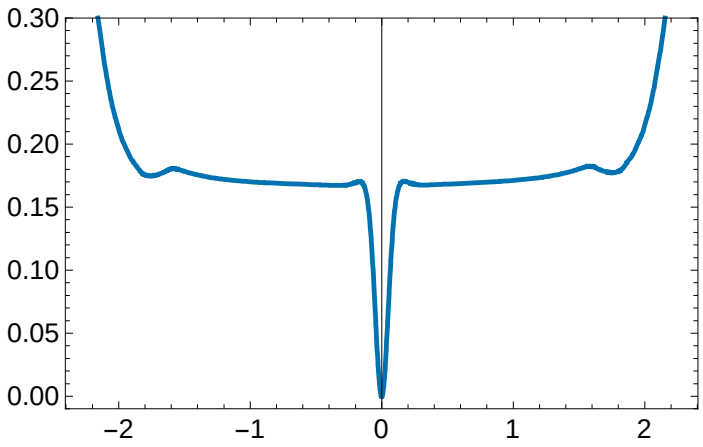
```
vMaskCenter[numPairs_] := Array[s, 2 numPairs + 1];
```

This generates the trapping solution

```
solutionCenter =
 With[{numPairs = 7}, NMinimize[WeightFunction[eMaskCenter[numPairs],
    numPairs, 1, targetvoltageCenter, 1], vMaskCenter[numPairs]]]
```

```
{0.000565488, {s[1] → -2.83974, s[2] → 0.00944954, s[3] → 0.00958674,
  s[4] → 0.12653, s[5] → 0.127328, s[6] → 0.81231, s[7] → 0.818129,
  s[8] → -3.6189, s[9] → -3.60324, s[10] → 0.813495, s[11] → 0.819311,
  s[12] → 0.126909, s[13] → 0.127704, s[14] → 0.00948421, s[15] → 0.0096203}}
```

```
solutionFrame[solutionCenter, eMaskCenter[7], 1]
```




```
Transpose[{AllElectrodeNames, targetvoltageCenter[7], eMaskCenter[7]}]
```

```
{{GND, 0, 0}, {RF, 0, 0}, {G01, 0, 0}, {G02, 0, 0}, {G03, 0, 0}, {G04, 0, 0}, {G05, 0, 0},
 {G06, 0, 0}, {G07, 0, 0}, {G08, 0, 0}, {L01, 0, 0}, {L02, 0, 0}, {L03, 0, 0}, {L04, 0, 0},
 {L05, 0, 0}, {L06, 0, 0}, {L07, 0, 0}, {L08, 0, 0}, {L09, 0, 0}, {L10, 0, 0},
 {L11, 0, 0}, {L12, 0, 0}, {L13, 0, 0}, {L14, 0, 0}, {L15, 0, 0}, {L16, 0, 0},
 {Q01, 0, 0}, {Q02, 0, 0}, {Q03, 0, 0}, {Q04, 0, 0}, {Q05, 0, 0}, {Q06, 0, 0},
 {Q07, 0, 0}, {Q08, 0, 0}, {Q09, 0, 0}, {Q10, 0, 0}, {Q11, 0, 0}, {Q12, 0, 0},
 {Q13, 0, s[2]}, {Q14, 0, s[3]}, {Q15, 0, s[4]}, {Q16, 0, s[5]}, {Q17, 0, s[6]},
 {Q18, 0, s[7]}, {Q19, 0, s[8]}, {Q20, 0, s[9]}, {Q21, 0, s[10]}, {Q22, 0, s[11]},
 {Q23, 0, s[12]}, {Q24, 0, s[13]}, {Q25, 0, s[14]}, {Q26, 0, s[15]}, {Q27, 0, 0},
 {Q28, 0, 0}, {Q29, 0, 0}, {Q30, 0, 0}, {Q31, 0, 0}, {Q32, 0, 0}, {Q33, 0, 0},
 {Q34, 0, 0}, {Q35, 0, 0}, {Q36, 0, 0}, {Q37, 0, 0}, {Q38, 0, 0}, {Q39, -1, s[1]},
 {Q40, -1, s[1]}, {T1, 0, 0}, {T2, 0, 0}, {T3, 0, 0}, {T4, 0, 0}, {T5, 0, 0}, {T6, 0, 0},
 {Y01, 0, 0}, {Y02, 0, 0}, {Y03, 0, 0}, {Y04, 0, 0}, {Y05, 0, 0}, {Y06, 0, 0},
 {Y07, 0, 0}, {Y08, 0, 0}, {Y09, 0, 0}, {Y10, 0, 0}, {Y11, 0, 0}, {Y12, 0, 0},
 {Y13, 0, 0}, {Y14, 0, 0}, {Y15, 0, 0}, {Y16, 0, 0}, {Y17, 0, 0}, {Y18, 0, 0},
 {Y19, 0, 0}, {Y20, 0, 0}, {Y21, 0, 0}, {Y22, 0, 0}, {Y23, 0, 0}, {Y24, 0, 0}}
```

Trapping Solutions positive outside target

Define target voltages for position along the nodal lines. This can be used to suggest trapping solutions that have a deeper trap depth or other intended properties.

```
targetvoltageCenter[numPairs_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]],
    ConstantArray[{1, 1}, Floor[numPairs / 2] - 1], {0, 0}, {0, 0}, {0, 0},
    ConstantArray[{1, 1}, Floor[numPairs / 2] - 1], ConstantArray[
     {0, 0}, 10 - Ceiling[numPairs / 2]], -1, -1, ConstantArray[0, 30]}];
Length[targetvoltageCenter[7]]
```

96

The electrodeMask defines the electrodes for which the voltage is to be adjusted to generate the trap.

```
eMaskCenter[numPairs_] := Flatten[{ConstantArray[0, 16], 0, 0, ConstantArray[0, 6],
    0, 0, ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]], Array[s, 2 numPairs, 2],
    ConstantArray[{0, 0}, 9 - Floor[numPairs / 2]], s[1], s[1], ConstantArray[0, 30]}];
```

Here we just define the number of degrees of freedom to match those in electrodeMask.

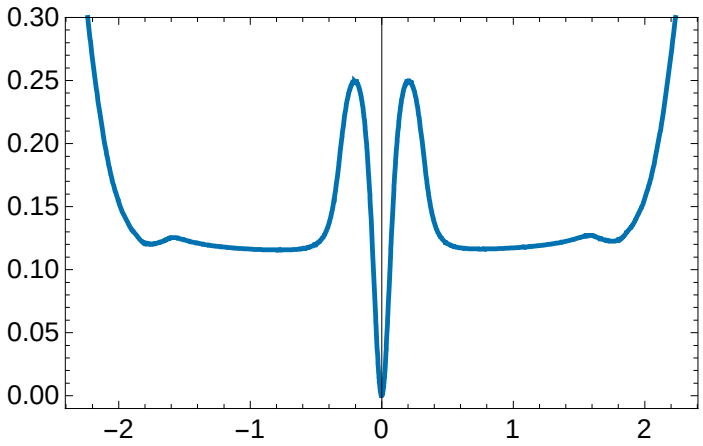
```
vMaskCenter[numPairs_] := Array[s, 2 numPairs + 1];
```

This generates the trapping solution

```
solutionCenter =
 With[{numPairs = 9}, NMinimize[WeightFunction[eMaskCenter[numPairs],
    numPairs, 1, targetvoltageCenter, 1], vMaskCenter[numPairs]]]
```

```
{0.000429905,
 {s[1] → -2.57011, s[2] → 1.00084, s[3] → 1.00087, s[4] → 1.00825, s[5] → 1.00837,
  s[6] → 1.11015, s[7] → 1.11088, s[8] → 0.703571, s[9] → 0.70889, s[10] → -3.17232,
  s[11] → -3.15801, s[12] → 0.704581, s[13] → 0.709897, s[14] → 1.11049,
  s[15] → 1.11121, s[16] → 1.00827, s[17] → 1.0084, s[18] → 1.00084, s[19] → 1.00087}}
```

```
solutionFrame[solutionCenter, eMaskCenter[9], 1]
```



Rotation Solutions

Define target voltages for position along the nodal lines. This can be used to suggest trapping solutions that have a deeper trap depth or other intended properties.

```
targetvoltageCenter[numPairs_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]],
    ConstantArray[{1, -1}, Floor[numPairs / 2]], {1, -1},
    ConstantArray[{1, -1}, Floor[numPairs / 2]], ConstantArray[{0, 0},
     10 - Ceiling[numPairs / 2]], -1, -1, ConstantArray[0, 30]}];
Length[targetvoltageCenter[7]]
```

96

The electrodeMask defines the electrodes for which the voltage is to be adjusted to generate the trap.

```
eMaskCenter[numPairs_] := Flatten[{ConstantArray[0, 16], 0, 0,
    ConstantArray[0, 6], 0, 0, ConstantArray[{0, 0}, 10 - Ceiling[numPairs / 2]],
    Array[s, 2 numPairs, 2], ConstantArray[{0, 0}, 9 - Floor[numPairs / 2]],
    -s[1], s[1], ConstantArray[0, 30]}];
```

Here we just define the number of degrees of freedom to match those in electrodeMask.

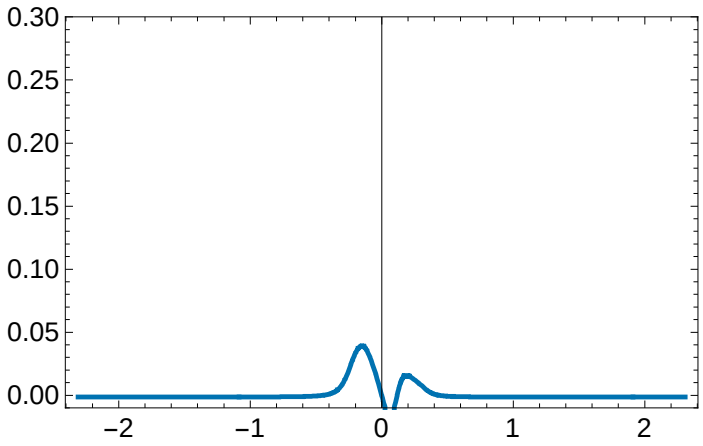
```
vMaskCenter[numPairs_] := Array[s, 2 numPairs + 1];
```

This generates the trapping solution

```
asymmetry[v_] := Sum[(v[[i]] + v[[i + 1]])^2, {i, 3, 95, 2}]

WeightFunction[v_, numPairs_, freq_, targetvoltage_, region_] := With[{posx = 0},
  Abs[(freqWeight (CharacterizeAxialFrequency[DCPotential[x, y, z, v, region],
          {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu] / 10^6)^2 +
     fieldWeight ((D[DCPotential[x, y, z, v, region], z])^2 +
        (D[DCPotential[x, y, z, v, region], y])^2 +
        (D[DCPotential[x, y, z, v, region], x])^2) + symWeight asymmetry[v] +
     eWeight Total[((v - targetvoltage[numPairs]) (v - targetvoltage[numPairs])).
        electrodepenalty[posx, region]]) /.
    {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}]]

solutionCenter =
 With[{numPairs = 9}, NMinimize[WeightFunction[eMaskCenter[numPairs],
    numPairs, 1, targetvoltageCenter, 1], vMaskCenter[numPairs]]]
```

$$\{7.99224 \times 10^{-9}, \{s[1] \to 0.0780514, s[2] \to 0.318958, s[3] \to 0.0207163,$$
$$s[4] \to 0.0152942, s[5] \to 0.160838, s[6] \to -0.199346, s[7] \to 0.41571,$$
$$s[8] \to 0.0347557, s[9] \to 0.0313313, s[10] \to 0.114899, s[11] \to 0.199245,$$
$$s[12] \to 0.315206, s[13] \to 0.971981, s[14] \to 0.136044, s[15] \to -0.522917,$$
$$s[16] \to 0.015543, s[17] \to -0.245148, s[18] \to -0.51909, s[19] \to 0.308937\}\}$$

```
solutionFrame[solutionCenter, eMaskCenter[9], 1]
```



CompensationSolutions

Handcrafted Solutions for central Quantum Region

Function to plot a solution

```
solutionFrame[sol_, region_] := With[{bundle = Evaluate[
      DCPotential[myx, nodeY[myx, region], nodeZ[myx, region], sol, region]]},
  With[{minpos = 0}, With[{minvalue = DCPotential[minpos, nodeY[minpos, region],
       nodeZ[minpos, region], sol, region]}, Plot[Evaluate[bundle - minvalue],
     {myx, xmin[region], xmax[region]}, PlotRange → {-0.01, 0.3}]]]]
```

Trapping Solutions

Ansatz for a trap at the center of the linear section (between Q19 and Q20). To rotate principal axes, add the rotation solution calculated below. Only one electrode pair is set to a negative voltage, the outer control electrodes are used to cancel electric fields at the rf node. All other electrodes are at ground.

```
centralAnsatz = Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 9],
    {a, b}, ConstantArray[{0, 0}, 9], -2, -2, ConstantArray[0, 30]}];
Length[centralAnsatz]
```

96

Solve by canceling lateral and vetical fields at rf node

```
centralsol = centralAnsatz /. Flatten[
   With[{posx = 0, r = 1}, Solve[{(D[DCPotential[x, y, z, centralAnsatz, r], z] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
      (D[DCPotential[x, y, z, centralAnsatz, r], y] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]]
```

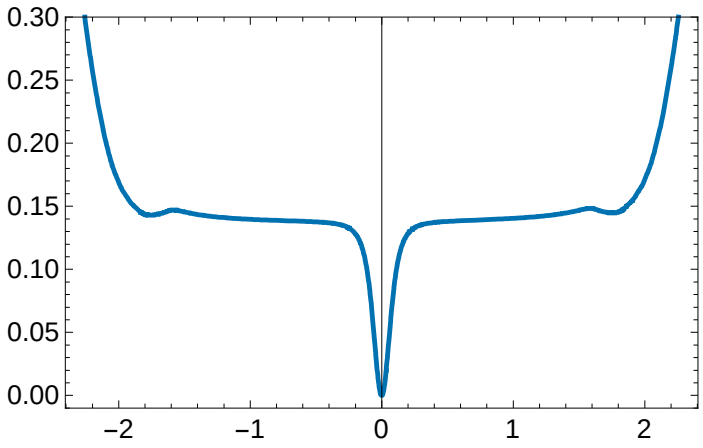
```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -2.19977, -2.1854,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -2, -2, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

Calculate axial trap frequency

```
With[{region = 1, posx = 0},
 CharacterizeAxialFrequency[DCPotential[x, y, z, centralsol, region],
  {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu]]
```

```
735 962.
```

```
solutionFrame[centralsol, 1]
```



For this ansatz, we are still putting -1 volt on the central electrode pair, but we are assigning +1V to 5 electrode pairs on either side

```
centralAnsatz2 = Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 4],
    ConstantArray[{1, 1}, 5], {-1, -1}, ConstantArray[{1, 1}, 5],
    ConstantArray[{0, 0}, 4], a, b, ConstantArray[0, 30]}];
Length[centralAnsatz2]
```

```
96
```

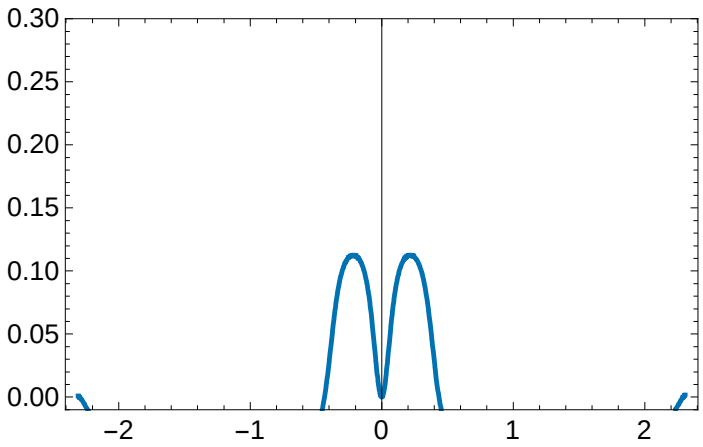
```
centralsol2 = centralAnsatz2 /. Flatten[
   With[{posx = 0, r = 1}, Solve[{(D[DCPotential[x, y, z, centralAnsatz2, r], z] /.
          {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
       (D[DCPotential[x, y, z, centralAnsatz2, r], y] /.
          {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1,
 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, -0.44526, -0.442245, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

```
With[{region = 1, posx = 0},
 CharacterizeAxialFrequency[DCPotential[x, y, z, centralsol2, region],
  {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu]]
```

```
701114.
```

```
solutionFrame[centralsol2, 1]
```




```
centralAnsatz3 = Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 6],
    ConstantArray[{0, 0}, 3], {-5, -5}, ConstantArray[{0, 0}, 3],
    ConstantArray[{0, 0}, 6], a, b, ConstantArray[0, 30]}];
Length[centralAnsatz3]
```

```
96
```

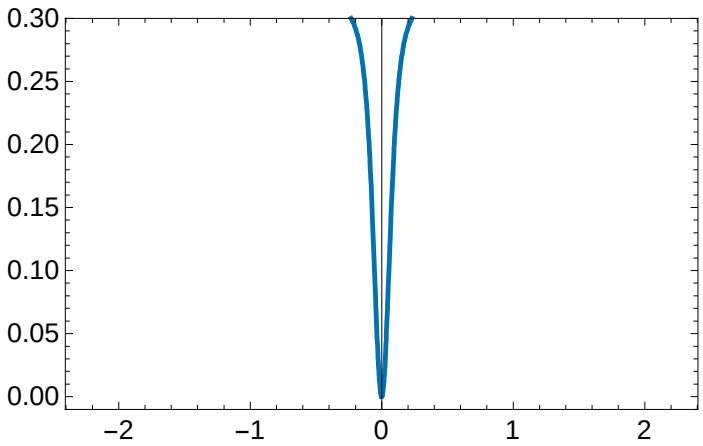
```
centralsol3 = centralAnsatz3 /. Flatten[
   With[{posx = 0, r = 1}, Solve[{(D[DCPotential[x, y, z, centralAnsatz3, r], z] /.
          {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
       (D[DCPotential[x, y, z, centralAnsatz3, r], y] /.
          {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -5, -5, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -4.57082, -4.55094, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

```
With[{region = 1, posx = 0},
 CharacterizeAxialFrequency[DCPotential[x, y, z, centralsol3, region],
  {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu]]
```

1.11138×10^6

```
solutionFrame[centralsol3, 1]
```




```
centralAnsatz4 = Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 4],
    ConstantArray[{1, 1}, 5], {-1, -1}, ConstantArray[{1, 1}, 5],
    ConstantArray[{0, 0}, 4], a, b, ConstantArray[0, 30]}];
Length[centralAnsatz3]
```

96

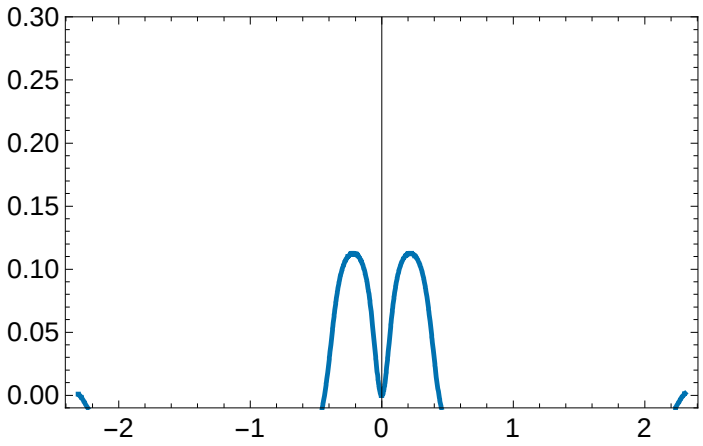
```
centralsol4 = centralAnsatz4 /. Flatten[
   With[{posx = 0, r = 1}, Solve[{(D[DCPotential[x, y, z, centralAnsatz4, r], z] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
       (D[DCPotential[x, y, z, centralAnsatz4, r], y] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1,
 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, -0.44526, -0.442245, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

```
With[{region = 1, posx = 0},
 CharacterizeAxialFrequency[DCPotential[x, y, z, centralsol4, region],
  {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu]]
```

701114.

```
solutionFrame[centralsol4, 1]
```



Rotation Solutions

```
ansatzN[numpairs_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[numpairs / 2]],
    ConstantArray[{a, -b}, numpairs],
    ConstantArray[{0, 0}, 9 - Floor[numpairs / 2]], -1, 1, ConstantArray[0, 30]}];
Length[
 ansatzN[
  4]]
```

```
96
```

```
With[{r = 1, posx = 0, numpairs = 5},
 Solve[{(D[DCPotential[x, y, z, ansatzN[numpairs], r], z] /.
       {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
   (D[DCPotential[x, y, z, ansatzN[numpairs], r], y] /.
       {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]
```

```
{{a → 0.898219, b → 0.891353}}
```

Compensation Solutions

```
ansatzOuter = Flatten[
   {ConstantArray[0, 26], ConstantArray[{0, 0}, 19], -a, -b, ConstantArray[0, 30]}];
Length[
 ansatzN[
  4]]
```

```
96
```

```
ansatzInner[numpairs_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[numpairs / 2]],
    ConstantArray[{a, b}, numpairs],
    ConstantArray[{0, 0}, 9 - Floor[numpairs / 2]], 0, 0, ConstantArray[0, 30]}];
Length[
 ansatzN[
  4]]
```

```
96
```

Vertical

```
With[{r = 1, posx = 0},
 Solve[{(D[DCPotential[x, y, z, ansatzOuter, r], z] /. {x → posx, y → nodeY[posx, r],
       z → nodeZ[posx, r]}) == -1, (D[DCPotential[x, y, z, ansatzOuter, r], y] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]
```

```
{{a → 0.776914, b → 0.774764}}
```

```
With[{r = 1, posx = 0, numpairs = 5},
 Solve[{(D[DCPotential[x, y, z, ansatzInner[numpairs], r], z] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == -1,
   (D[DCPotential[x, y, z, ansatzInner[numpairs], r], y] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0}, {a, b}]]
```

```
{{a → 0.532323, b → 0.531753}}
```

Horizontal

```
With[{r = 1, posx = 0},
 Solve[{(D[DCPotential[x, y, z, ansatzOuter, r], z] /. {x → posx, y → nodeY[posx, r],
       z → nodeZ[posx, r]}) == 0, (D[DCPotential[x, y, z, ansatzOuter, r], y] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 1}, {a, b}]]
```

```
{{a → -0.986601, b → 0.98492}}
```

```
With[{r = 1, posx = 0, numpairs = 5},
 Solve[{(D[DCPotential[x, y, z, ansatzInner[numpairs], r], z] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
   (D[DCPotential[x, y, z, ansatzInner[numpairs], r], y] /.
      {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 1}, {a, b}]]
```

```
{{a → 0.884851, b → -0.879236}}
```

Axial

```
ansatzAxial[distance_] :=
  Flatten[{ConstantArray[0, 26], ConstantArray[{0, 0}, 10 - Ceiling[distance / 2]],
    {a, a}, ConstantArray[{0, 0}, distance - 2], {-a, -a},
    ConstantArray[{0, 0}, 9 - Floor[distance / 2]], c, c2, ConstantArray[0, 30]}];
Length[ansatzAxial[
  4]]
```

```
96
```

```
MatrixForm[{AllElectrodeNames, ansatzAxial[3]}]
```

GND	RF	G01	G02	G03	G04	G05	G06	G07	G08	L01	L02	L03	L04	L05	L06	L07	L08	L09	L10
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

```
axialsol = Flatten[With[{r = 1, posx = 0, distance = 3},
   Solve[{(D[DCPotential[x, y, z, ansatzAxial[distance], r], z] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
      (D[DCPotential[x, y, z, ansatzAxial[distance], r], y] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == 0,
      (D[DCPotential[x, y, z, ansatzAxial[distance], r], x] /.
         {x → posx, y → nodeY[posx, r], z → nodeZ[posx, r]}) == -1}, {a, c, c2}]]]
```

```
{a → 0.962952, c → 0.000262614, c2 → -0.000270445}
```

Shuttling

Here, we are calculating shuttling solutions without rotating the principal axes.

Shuttling Prerequisites

General definitions for shuttling. ElectrodeNames give the names of the electrodes in the same order as the voltage solution and is used for printing the solutions in a file.

```
ElectrodeNames = Flatten[{Table["G" <> IntegerString[i, 10, 2], {i, 1, 8}],
   Table["L" <> IntegerString[i, 10, 2], {i, 1, 16}],
   Table["Q" <> IntegerString[i, 10, 2], {i, 1, 40}],
   Table["T" <> IntegerString[i, 10, 1], {i, 1, 6}],
   Table["Y" <> IntegerString[i, 10, 2], {i, 1, 24}]}]
```

```
{G01, G02, G03, G04, G05, G06, G07, G08, L01, L02, L03, L04, L05, L06, L07, L08,
 L09, L10, L11, L12, L13, L14, L15, L16, Q01, Q02, Q03, Q04, Q05, Q06, Q07, Q08,
 Q09, Q10, Q11, Q12, Q13, Q14, Q15, Q16, Q17, Q18, Q19, Q20, Q21, Q22, Q23,
 Q24, Q25, Q26, Q27, Q28, Q29, Q30, Q31, Q32, Q33, Q34, Q35, Q36, Q37, Q38,
 Q39, Q40, T1, T2, T3, T4, T5, T6, Y01, Y02, Y03, Y04, Y05, Y06, Y07, Y08, Y09,
 Y10, Y11, Y12, Y13, Y14, Y15, Y16, Y17, Y18, Y19, Y20, Y21, Y22, Y23, Y24}
```

```
AllElectrodeNames = Flatten[{"GND", "RF", ElectrodeNames}];
```

globalProcess variable is used to show progress on the sometimes slow calculations. There is only one

global variable and all Progress bars will move no matter which of the expressions that support the progress bar are used.

```
globalProgress = 0;
```

This function plots the axial trapping potential for each solution in sol.

```
solutionFrames[sol_, eMask_, region_] :=
 With[{bundle = Evaluate[Table[(globalProgress = i / Length[sol];
       DCPotential[myx, nodeY[myx, region], nodeZ[myx, region],
        eMask[sol[[i, 1]]] /. sol[[i, 3]], region]), {i, 1, Length[sol]}]]},
  Table[(globalProgress = i / Length[sol];
    With[{minpos = sol[[i, 1]]}, With[
      {minvalue = DCPotential[minpos, nodeY[minpos, region], nodeZ[minpos, region],
         eMask[sol[[i, 1]]] /. sol[[i, 3]], region]}, Plot[Evaluate[bundle[[i]] -
         minvalue], {myx, xmin[region], xmax[region]}, PlotRange → {-0.01, 0.3},
       PlotLabel → "Frame " <> ToString[i] <> " x=" <> ToString[sol[[i, 1]]],
       Epilog → {Arrow[{{minpos, 0.1}, {minpos, 0.01}}]}]]]), {i, 1, Length[sol]}]]
```

Write the solution to a file.

```
writeSolution[filename_, sol_, eMask_, region_] := Module[{stream},
  stream = OpenWrite[filename];
  Export[stream, {ElectrodeNames}, "Table"];
  WriteString[stream, "\n"];
  Export[stream, Table[
    eMask[sol[[i, 1]]][[3 ;; 96]] /. sol[[i, 3]], {i, 1, Length[sol]}], "Table"];
  Close[stream];]
```

Weights for the different contributions to the WeightFunction

```
{freqWeight, fieldWeight, symWeight, eWeight} = {100, 1, 10^-4, 10^-5}
```

$$\left\{100,\ 1,\ \frac{1}{10\,000},\ \frac{1}{100\,000}\right\}$$

Calculate the length of the field vector generated by each electrode at the node at position posx along the x - axis. The inverse of this quantity is then used as an additional weight to have the solution favor those electrodes that generate the strongest field at the given position. Or, in other words, prevent high voltages at far away electrodes.

```
electrodepenalty[posx_, region_] :=
 Table[With[{elec = DCPotential[x, y, z, UnitVector[96, e], region]},
   1 / Sqrt[(D[elec, z])^2 + (D[elec, y])^2 + (D[elec, x])^2 /.
      {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}]], {e, 1, 96}]
```

A weight for asymmetry of a solution to favor symmetric solutions. In case of rotating solutions this function is used for the deviation of the rotating solution.

```
asymmetry[v_] := Sum[(v[[i]] - v[[i + 1]])^2, {i, 3, 95, 2}]
```

The WeightFunction for the global optimization for a trap at axial position posx [mm] with trap frequency freq [MHz], targetvoltages targetvoltage [V] for a given region of the trap. Contributions are:
freqWeight: Favoring the requested axial trap frequency
fieldWeight: Favoring vanishing DC fields at the nodal location

symWeight: Favoring symmetric solutions
eWeight: Favoring voltages on nearby electrodes

```
WeightFunction[v_, posx_, freq_, targetvoltage_, region_] :=
 Abs[ (freqWeight ( CharacterizeAxialFrequency[DCPotential[x, y, z, v, region],
          {posx, nodeY[posx, region], nodeZ[posx, region]}, IonAmu] / 10^6 - freq)^2 +
     fieldWeight ((D[DCPotential[x, y, z, v, region], z])^2 +
        (D[DCPotential[x, y, z, v, region], y])^2 +
        (D[DCPotential[x, y, z, v, region], x])^2) + symWeight asymmetry[v] +
     eWeight Total[((v - targetvoltage[posx]) (v - targetvoltage[posx])).
        electrodepenalty[posx, region]]) /.
   {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}]
```

Generate plots fields in V/m and frequency

```
verificationPlots[sol_, eMask_, region_] :=
 {ListPlot[Table[(globalProgress = i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], x] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "x-Field"],
  ListPlot[Table[(globalProgress = 1 / 4 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], y] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "y-Field"],
  ListPlot[Table[(globalProgress = 1 / 2 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]},
      {posx, 1000 D[DCPotential[x, y, z, eMask[posx] /. sol[[i, 3]], region], z] /.
        {x → posx, y → nodeY[posx, region], z → nodeZ[posx, region]}}]), {i, 1,
     Length[sol]}], AxesLabel → {"Field V/m", "pos [mm]"}, PlotLabel → "z-Field"],
  ListPlot[Table[(globalProgress = 3 / 4 + i / (4 Length[sol]);
     With[{posx = sol[[i, 1]]}, {posx, CharacterizeAxialFrequency[DCPotential[x, y,
          z, eMask[posx] /. sol[[i, 3]], region], {posx, nodeY[posx, region],
          nodeZ[posx, region]}, IonAmu] / 10^6}]), {i, 1, Length[sol]}],
   AxesLabel → {"trap freq [MHz]", "pos [mm]"}, PlotLabel → "trap frequency"]}
```

Solution for central region

```
truncateToZero[x_] = If[x > 0,
  Floor[x],
  Ceiling[x]]
```

```
If[x > 0, Floor[x], Ceiling[x]]
```

Define target voltages for different positions along the nodal lines. This can be used to suggest trapping solutions that have a deeper trap depth or other intended properties.

```
targetvoltageR1[pos_] =
  Which[Abs[pos] < 1.88, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}],
   Abs[pos] < 1.975, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 4], 1, 1, ConstantArray[0, 24]}],
   Abs[pos] < 2.069, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12],
     -1, -1, 0, 0, 1, 1, ConstantArray[0, 26]}],
   Abs[pos] < 2.163, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12],
     -1, -1, 1, 1, ConstantArray[0, 28]}],
   Abs[pos] < 2.259, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12],
     -1, -1, ConstantArray[0, 28], 1, 1}],
   Abs[pos] < 2.309, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12],
     -1, -1, ConstantArray[0, 26], 1, 1, 0, 0}]];
```

The electrodeMask defines the electrodes for which the voltage is to be adjusted to generate the trap.

```
eMaskR1[xpos_] = Which[Abs[xpos] < 8 × 0.07,
   Flatten[{ConstantArray[0, 16], -1, -1, ConstantArray[0, 6],
     -1, -1, ConstantArray[0, 14 + 2 truncateToZero[xpos / 0.07]],
     Array[s, 10], ConstantArray[0, 14 - 2 truncateToZero[xpos / 0.07]],
     -2, -2, ConstantArray[0, 30]}],
   Abs[xpos] < 9 × 0.07, Flatten[{ConstantArray[0, 4], Array[s, 2],
     ConstantArray[0, 10], -1, -1, ConstantArray[0, 6], -1,
     -1, ConstantArray[0, If[xpos > 0, 30, 0]], Array[s, 8, 3],
     ConstantArray[0, If[xpos < 0, 30, 0]], -2, -2, ConstantArray[0, 30]}],
   Abs[xpos] < 10 × 0.07, Flatten[{ConstantArray[0, 2], Array[s, 4],
     ConstantArray[0, 4], ConstantArray[0, 6], -1, -1, ConstantArray[0, 6],
     -1, -1, ConstantArray[0, If[xpos > 0, 32, 0]], Array[s, 6, 5],
     ConstantArray[0, If[xpos < 0, 32, 0]], -2, -2, ConstantArray[0, 30]}],
   Abs[xpos] < 11 × 0.07, Flatten[{ConstantArray[0, 2], Array[s, 4],
     ConstantArray[0, 2], Array[s, 2, 5], ConstantArray[0, 6], -1, -1, ConstantArray[
      0, 6], -1, -1, ConstantArray[0, If[xpos > 0, 34, 0]], Array[s, 4, 7],
     ConstantArray[0, If[xpos < 0, 34, 0]], -2, -2, ConstantArray[0, 30]}],
   Abs[xpos] < 12 × 0.07, Flatten[{ConstantArray[0, 2], Array[s, 8],
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 6], -1,
```

```
    -1, ConstantArray[0, If[xpos > 0, 36, 0]], Array[s, 2, 9],
    ConstantArray[0, If[xpos < 0, 36, 0]], -2, -2, ConstantArray[0, 30]}],
  Abs[xpos] < 22 × 0.07, Flatten[{ConstantArray[0, 2], Array[s, 8],
    ConstantArray[0, 6], -1, -1, ConstantArray[0, 6], -1, -1,
    ConstantArray[0, 38], -2, -2, ConstantArray[0, 30]}],
  Abs[xpos] < 1.61, Flatten[{ConstantArray[0, 2], Array[s, 8], ConstantArray[0, 6],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2,
    -2, ConstantArray[0, 4], Array[s, 2, 9], ConstantArray[0, 24]}],
  Abs[xpos] < 1.692, Flatten[{ConstantArray[0, 2], Array[s, 6],
    ConstantArray[0, 8], -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38],
    -2, -2, ConstantArray[0, 2], Array[s, 4, 7], ConstantArray[0, 24]}],
  Abs[xpos] < 1.782, Flatten[{ConstantArray[0, 2], Array[s, 4],
    ConstantArray[0, 10], -1, -1, ConstantArray[0, 6], -1, -1,
    ConstantArray[0, 38], -2, -2, Array[s, 6, 5], ConstantArray[0, 24]}],
  Abs[xpos] < 1.88, Flatten[{ConstantArray[0, 2], Array[s, 2],
    ConstantArray[0, 12], -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38],
    -2, -2, Array[s, 6, 3], ConstantArray[0, 22], Array[s, 2, 9]}],
  Abs[xpos] < 1.975, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38],
    -2, -2, Array[s, 6], ConstantArray[0, 20], Array[s, 4, 7]}],
  Abs[xpos] < 2.069, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38],
    -2, -2, Array[s, 4], ConstantArray[0, 20], Array[s, 6, 5]}],
  Abs[xpos] < 2.163, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38],
    -2, -2, Array[s, 2], ConstantArray[0, 20], Array[s, 8, 3]}],
  Abs[xpos] < 2.259, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2,
    ConstantArray[0, 12], Array[s, 2], ConstantArray[0, 6], Array[s, 10, 3]}],
  Abs[xpos] < 2.309, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
    -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2,
    ConstantArray[0, 12], Array[s, 2], ConstantArray[0, 6], Array[s, 8, 3], 0, 0}]];
```

Here we just define the number of degrees of freedom to match those in electrodeMask.

```
vMaskR1[xpos_] = Which[Abs[xpos] < 12 × 0.07, Array[s, 10],
  Abs[xpos] < 22 × 0.07, Array[s, 8],
  Abs[xpos] < 2.163, Array[s, 10],
  Abs[xpos] < 2.259, Array[s, 12],
  Abs[xpos] < 2.309, Array[s, 10]];
```

This generates the trapping solution

```
solutionR1 = Table[Flatten[
    {posx, NMinimize[WeightFunction[eMaskR1[posx], posx, 0.5, targetvoltageR1, 1],
      vMaskR1[posx]]}, 1], {posx, xmin[1], xmax[1], 0.005}];
```

```
$Aborted
```

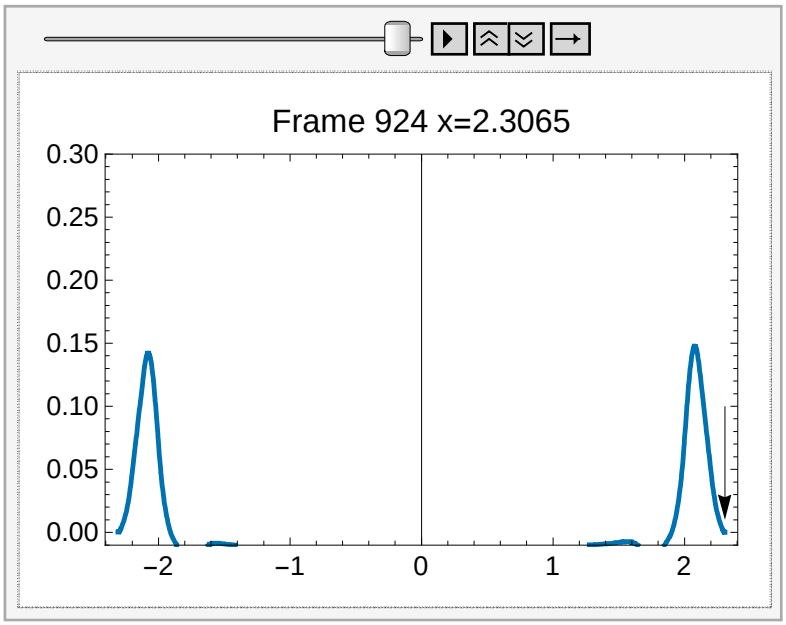
```
Length[solutionR1]
```

```
924
```

```
writeSolution["ShuttleR1.txt", solutionR1, eMaskR1, 1]
```

```
allFramesR1 = solutionFrames[solutionR1, eMaskR1, 1];
```

```
ListAnimate[allFramesR1, 5]
```



Write the animation in different formats

```
Export["shuttleR1.swf", allFramesR1, "FrameRate" → 5]
```

```
shuttleR1.swf
```

```
Export["shuttleR1.cdf", ListAnimate[allFramesR1, 5]]
```

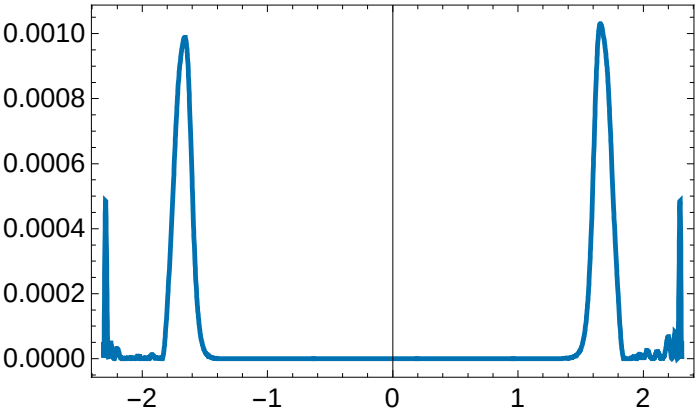
```
shuttleR1.cdf
```

```
Export["shuttleR1.png", allFramesR1[[1]]]
```

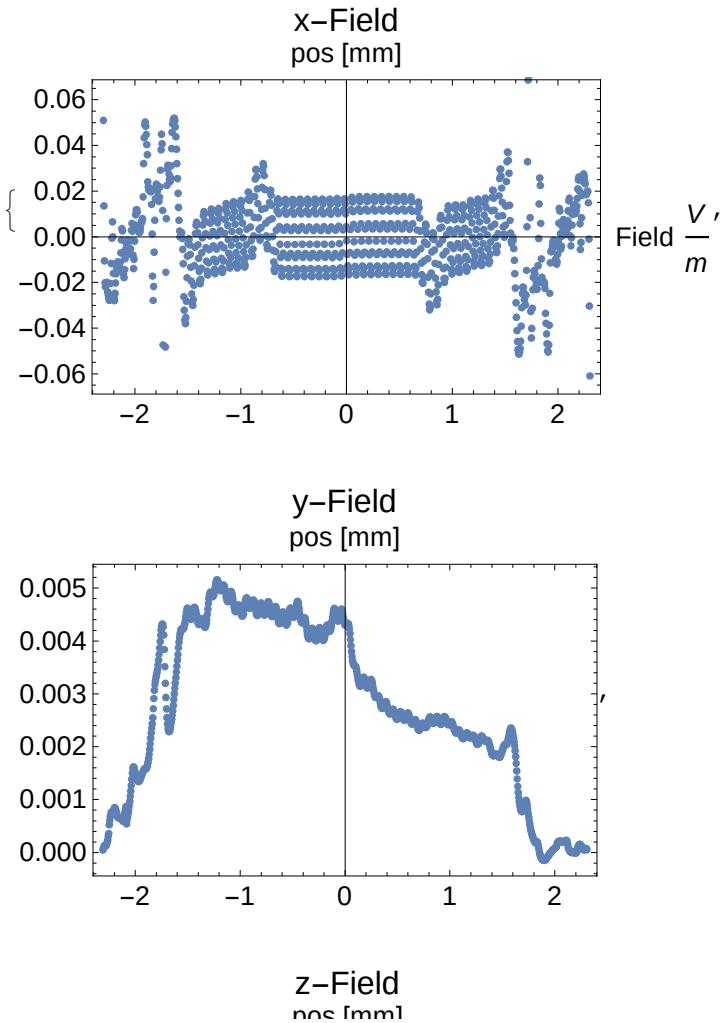
```
shuttleR1.png
```

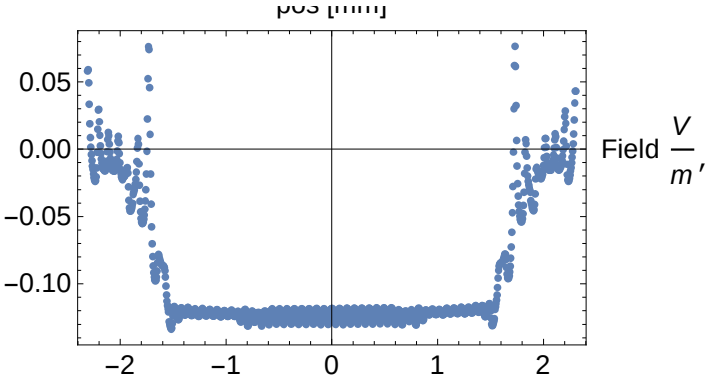
Verification

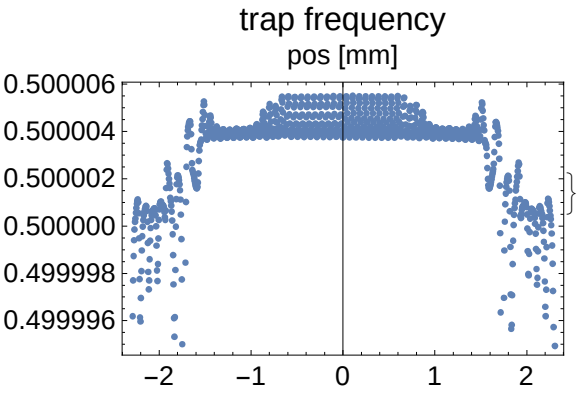
```
With[{region = 1},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```




```
verificationPlots[solutionR1, eMaskR1, 1]
```







Solution for loading leg L1 (top left)

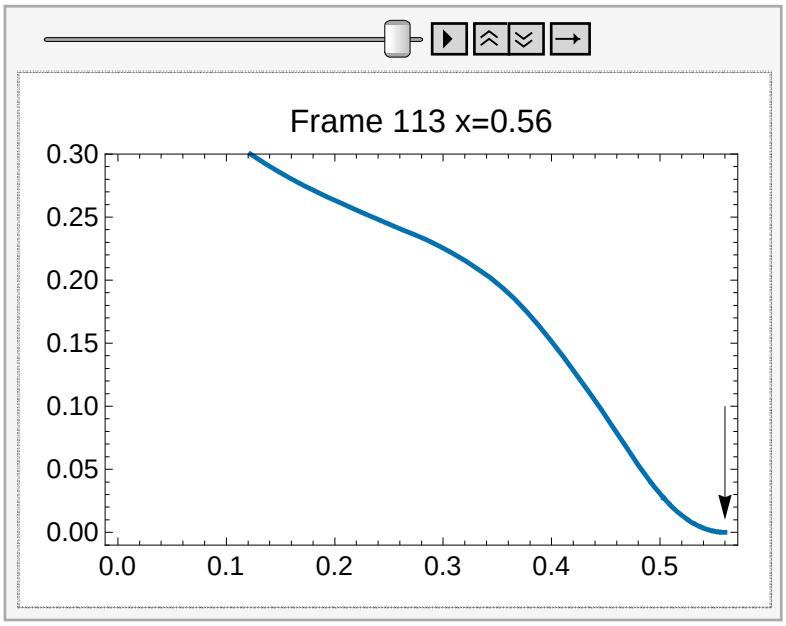
```
targetvoltageL1[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 2], 1, 1, ConstantArray[0, 20]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];

outerVoltage = -2;
outerVoltageL = -3;
```

```
eMaskL1[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 2],
     Array[s, 6], ConstantArray[0, 6], Array[s, 4, 7], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 14], Array[s, 2], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8, 3], ConstantArray[0, 10], ConstantArray[0, 6]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 12], Array[s, 4], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 6, 5], ConstantArray[0, 12], ConstantArray[0, 6]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 10], Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 4, 7], ConstantArray[0, 10], ConstantArray[0, 10]}],
   xpos < 0.650, Flatten[{ConstantArray[0, 10], Array[s, 6],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL1[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL1 = Table[Flatten[
    {posx, NMinimize[WeightFunction[eMaskL1[posx], posx, 0.5, targetvoltageL1, 2],
      vMaskL1[posx]]}, 1], {posx, xmin[2], xmax[2], 0.005}];

Length[solutionL1]
```

```
113
```

```
writeSolution["ShuttleL1.txt", solutionL1, eMaskL1, 1]

ProgressIndicator[Dynamic[globalProgress]]
```

```
allFramesL1 = solutionFrames[solutionL1, eMaskL1, 2];

ListAnimate[allFramesL1, 5]
```




```
Export["shuttleL1.swf", allFramesL1, "FrameRate" → 5]
```

```
shuttleL1.swf
```

```
Export["shuttleL1.cdf", ListAnimate[allFramesL1, 5]]
```

```
shuttleL1.cdf
```

```
Export["shuttleL1.png", allFramesL1[[1]]]
```

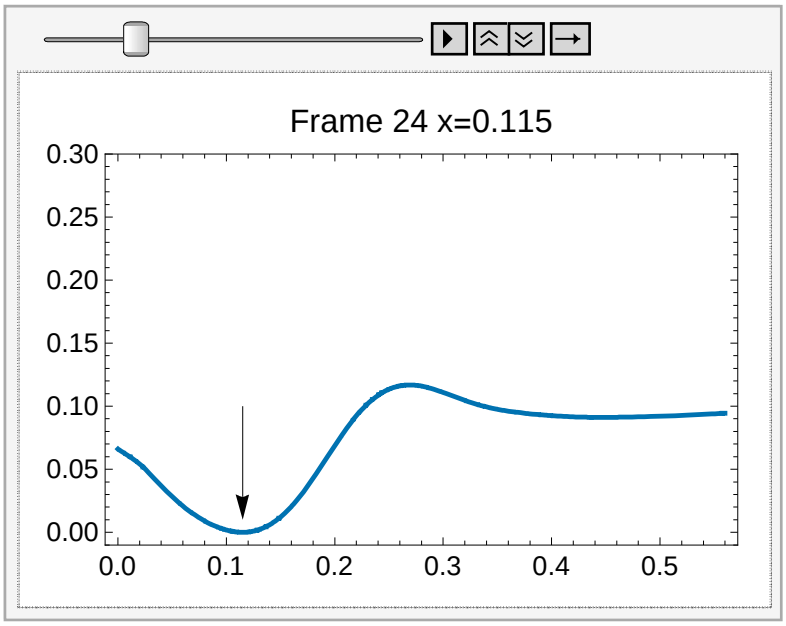
```
shuttleL1.png
```

Solution for loading leg L2 (bottom left)

```
targetvoltageL2[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 6],
     ConstantArray[0, 10], 1, 1, ConstantArray[0, 12]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];

outerVoltage = -2;
outerVoltageL = -3;
```

```
eMaskL2[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 6],
     Array[s, 2], ConstantArray[0, 2], Array[s, 8, 3], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 4], Array[s, 2], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 8, 3], ConstantArray[0, 8]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 2], Array[s, 4], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 6, 5], ConstantArray[0, 10]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     Array[s, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 8],
     Array[s, 4, 7], ConstantArray[0, 12]}], xpos < 0.650, Flatten[
    {ConstantArray[0, 16], outerVoltageL, outerVoltageL, Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL2[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL2 = Table[(globalProgress = (posx - xmin[3]) / (xmax[3] - xmin[3]);
    Flatten[{posx, NMinimize[WeightFunction[eMaskL2[posx], posx, 0.5,
        targetvoltageL2, 3], vMaskL2[posx]]}, 1]), {posx, xmin[3], xmax[3], 0.005}];

Length[solutionL2]
113

ProgressIndicator[Dynamic[globalProgress]]

writeSolution["ShuttleL2.txt", solutionL2, eMaskL2, 1]

allFramesL2 = solutionFrames[solutionL2, eMaskL2, 3];
```

```
ListAnimate[allFramesL2, 5]
```




```
Export["shuttleL2.swf", allFramesL2, "FrameRate" → 5]
```

```
shuttleL2.swf
```

```
Export["shuttleL2.cdf", ListAnimate[allFramesL2, 5]]
```

```
shuttleL2.cdf
```

```
Export["shuttleL2.png", allFramesL2[[1]]]
```

```
shuttleL2.png
```

Solution for loading leg L3 (bottom right)

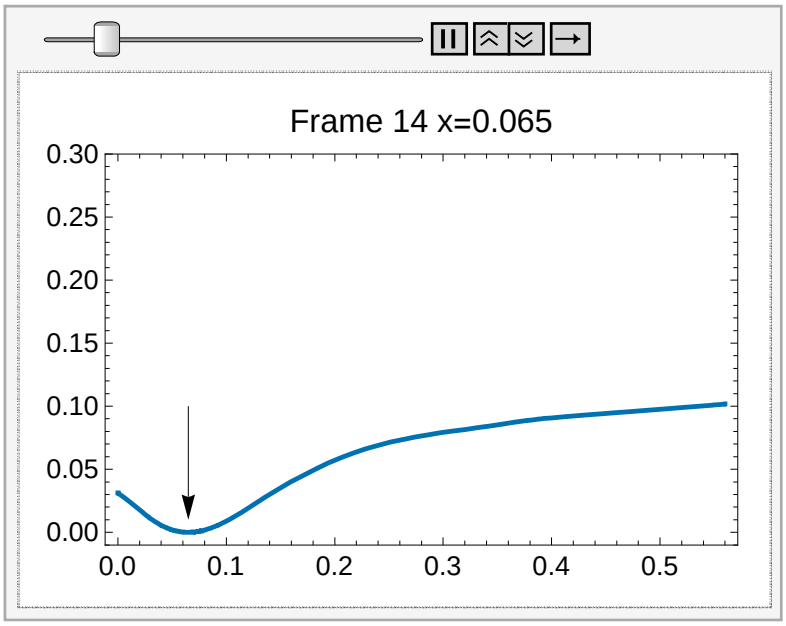
```
targetvoltageL3[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 6],
     ConstantArray[0, 10], 1, 1, ConstantArray[0, 12]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];
```

```
eMaskL3[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], -1, -1, ConstantArray[0, 6],
     -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6], ConstantArray[0, 6],
     Array[s, 2], ConstantArray[0, 2], Array[s, 8, 3], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     -1, -1, ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 16], -1, -1, ConstantArray[0, 4],
     Array[s, 2], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 8, 3], ConstantArray[0, 8]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 16], -1, -1, ConstantArray[0, 2],
     Array[s, 4], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 6, 5], ConstantArray[0, 10]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 16], -1, -1, Array[s, 6],
     -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 4, 7], ConstantArray[0, 12]}],
   xpos < 0.650, Flatten[{ConstantArray[0, 16], -1, -1, Array[s, 6],
     -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL3[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL3 = Table[Flatten[
    {posx, NMinimize[WeightFunction[eMaskL3[posx], posx, 0.5, targetvoltageL3, 4],
      vMaskL3[posx]]}, 1], {posx, xmin[4], xmax[4], 0.005}];

Length[solutionL3]
```

```
113
```

```
writeSolution["ShuttleL3.txt", solutionL3, eMaskL3, 1]

allFramesL3 = solutionFrames[solutionL3, eMaskL3, 4];
```

```
ListAnimate[allFramesL3, 5]
```




```
Export["shuttleL3.swf", allFramesL3, "FrameRate" → 5]
```

```
shuttleL3.swf
```

```
Export["shuttleL3.cdf", ListAnimate[allFramesL3, 5]]
```

```
shuttleL3.cdf
```

```
Export["shuttleL3.png", allFramesL3[[1]]]
```

```
shuttleL3.png
```

Solution for loading leg L4 (top right)

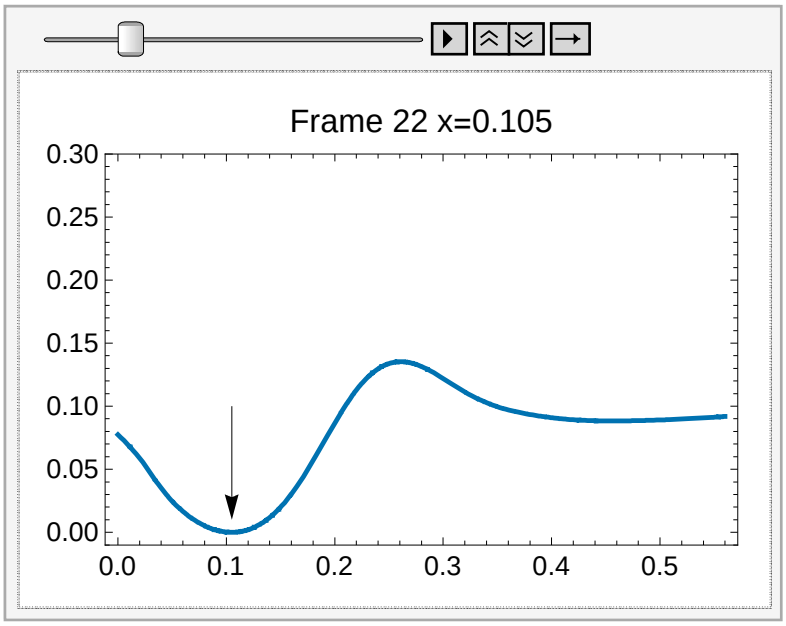
```
targetvoltageL4[pos_] =
  Which[pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];
```

```
eMaskL4[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], -1, -1, ConstantArray[0, 6],
     -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6], ConstantArray[0, 2],
     Array[s, 6], ConstantArray[0, 6], Array[s, 4, 7], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 14], Array[s, 2], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     Array[s, 8, 3], ConstantArray[0, 10], ConstantArray[0, 6]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 12], Array[s, 4], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     Array[s, 6, 5], ConstantArray[0, 12], ConstantArray[0, 6]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 10], Array[s, 6], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2, ConstantArray[0, 6],
     Array[s, 4, 7], ConstantArray[0, 10], ConstantArray[0, 10]}],
   xpos < 0.650, Flatten[{ConstantArray[0, 10], Array[s, 6], -1, -1,
     ConstantArray[0, 6], -1, -1, ConstantArray[0, 38], -2, -2,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL4[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL4 = Table[Flatten[
    {posx, NMinimize[WeightFunction[eMaskL4[posx], posx, 0.5, targetvoltageL4, 5],
      vMaskL4[posx]]}, 1], {posx, xmin[5], xmax[5], 0.005}];

Length[solutionL4]
```

```
113
```

```
writeSolution["ShuttleL4.txt", solutionL4, eMaskL4, 1]

allFramesL4 = solutionFrames[solutionL4, eMaskL4, 5];
```

```
ListAnimate[allFramesL4, 5]
```




```
Export["shuttleL4.swf", allFramesL4, "FrameRate" → 5]
```

```
shuttleL4.swf
```

```
Export["shuttleL4.cdf", ListAnimate[allFramesL4, 5]]
```

```
shuttleL4.cdf
```

```
Export["shuttleL4.png", allFramesL4[[1]]]
```

```
shuttleL4.png
```

Shuttling with Rotation

Solution for Central Region with rotation

```
targetvoltageR1Rot[pos_] := targetvoltageR1[pos] + rotationVoltages
eMaskR1Rot[pos_] := eMaskR1[pos] + rotationVoltages
```

```
solutionR1Rot =
  Table[(globalProgress = (posx - xmin[1]) / (xmax[1] - xmin[1]); Flatten[{posx,
      NMinimize[WeightFunction[eMaskR1Rot[posx], posx, 0.5, targetvoltageR1Rot, 1],
       vMaskR1[posx]]}, 1]), {posx, xmin[1], xmax[1], 0.005}];
```

```
$Aborted
```

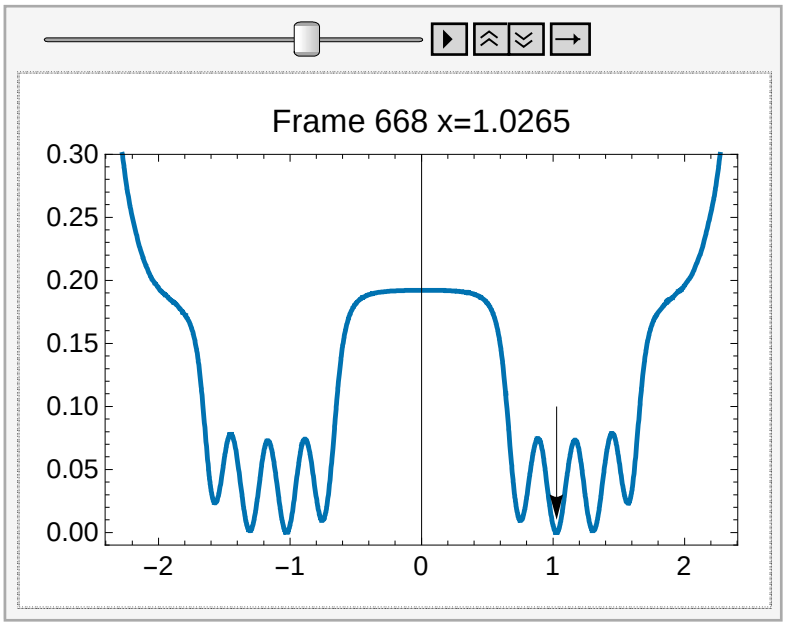
```
writeSolution["ShuttleR1Rot.txt", solutionR1Rot, eMaskR1Rot, 1]
```

```
ProgressIndicator[Dynamic[globalProgress]]
```

```
allFramesR1Rot = solutionFrames[solutionR1Rot, eMaskR1Rot, 1];
```

)

```
ListAnimate[allFramesR1Rot, 5]
```




```
Export["shuttleR1Rot.swf", allFramesR1Rot, "FrameRate" → 5]
```

shuttleR1.swf

```
Export["shuttleR1Rot.cdf", ListAnimate[allFramesR1Rot, 5]]
```

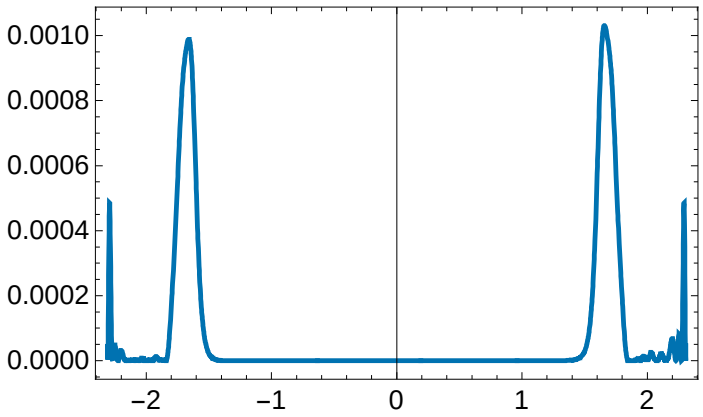
shuttleR1.cdf

```
Export["shuttleR1Rot.png", allFramesR1Rot[[1]]]
```

shuttleR1.png

Verification

```
With[{region = 1},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```




```
verificationPlots[solutionR1Rot, eMaskR1Rot, 1]
```

```
$Aborted
```

Solution for loading leg L1 (top left) with rotation

```
targetvoltageL1Rot[pos_] := targetvoltageL1[pos] + rotationVoltages
eMaskL1Rot[pos_] := eMaskL1[pos] + rotationVoltages
```

```
targetvoltageL1[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 2], 1, 1, ConstantArray[0, 20]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];
```

```
outerVoltage = -2;
outerVoltageL = -3;
```

```
eMaskL1[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 2],
     Array[s, 6], ConstantArray[0, 6], Array[s, 4, 7], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 14], Array[s, 2], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8, 3], ConstantArray[0, 10], ConstantArray[0, 6]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 12], Array[s, 4], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 6, 5], ConstantArray[0, 12], ConstantArray[0, 6]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 10], Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 4, 7], ConstantArray[0, 10], ConstantArray[0, 10]}],
   xpos < 0.650, Flatten[{ConstantArray[0, 10], Array[s, 6],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL1[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL1Rot = Table[(globalProgress = (posx - xmin[2]) / (xmax[2] - xmin[2]);
    Flatten[{posx,
      NMinimize[WeightFunction[eMaskL1Rot[posx], posx, 0.5, targetvoltageL1Rot, 2],
       vMaskL1[posx]]}, 1]), {posx, xmin[2], xmax[2], 0.005}];

Length[solutionL1]
```

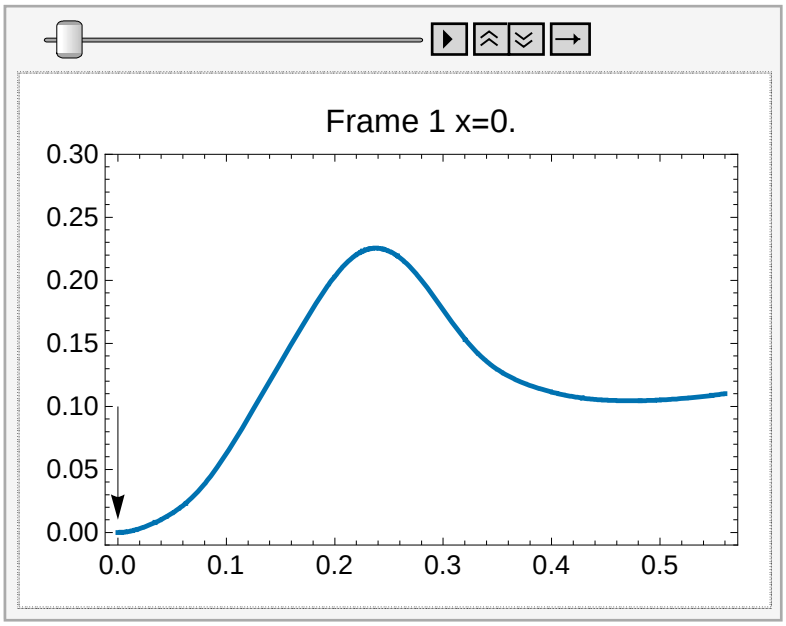
```
113
```

```
writeSolution["ShuttleL1Rot.txt", solutionL1Rot, eMaskL1Rot, 1]
```

```
ProgressIndicator[Dynamic[globalProgress]]
```

```
allFramesL1Rot = solutionFrames[solutionL1Rot, eMaskL1Rot, 2];
```

```
ListAnimate[allFramesL1Rot, 5]
```




```
Export["shuttleL1.swf", allFramesL1, "FrameRate" → 5]
```

```
shuttleL1.swf
```

```
Export["shuttleL1.cdf", ListAnimate[allFramesL1, 5]]
```

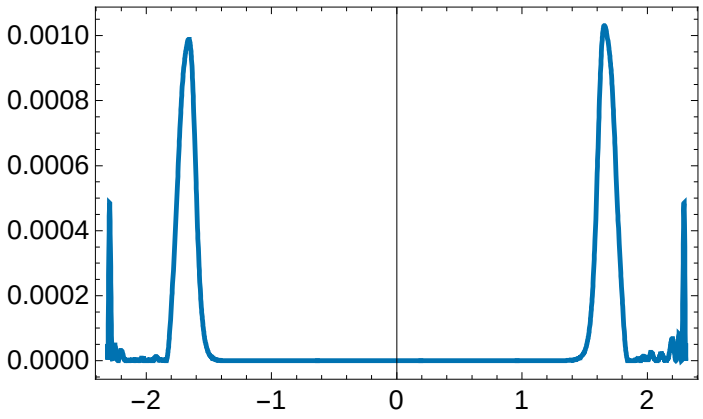
```
shuttleL1.cdf
```

```
Export["shuttleL1.png", allFramesL1[[1]]]
```

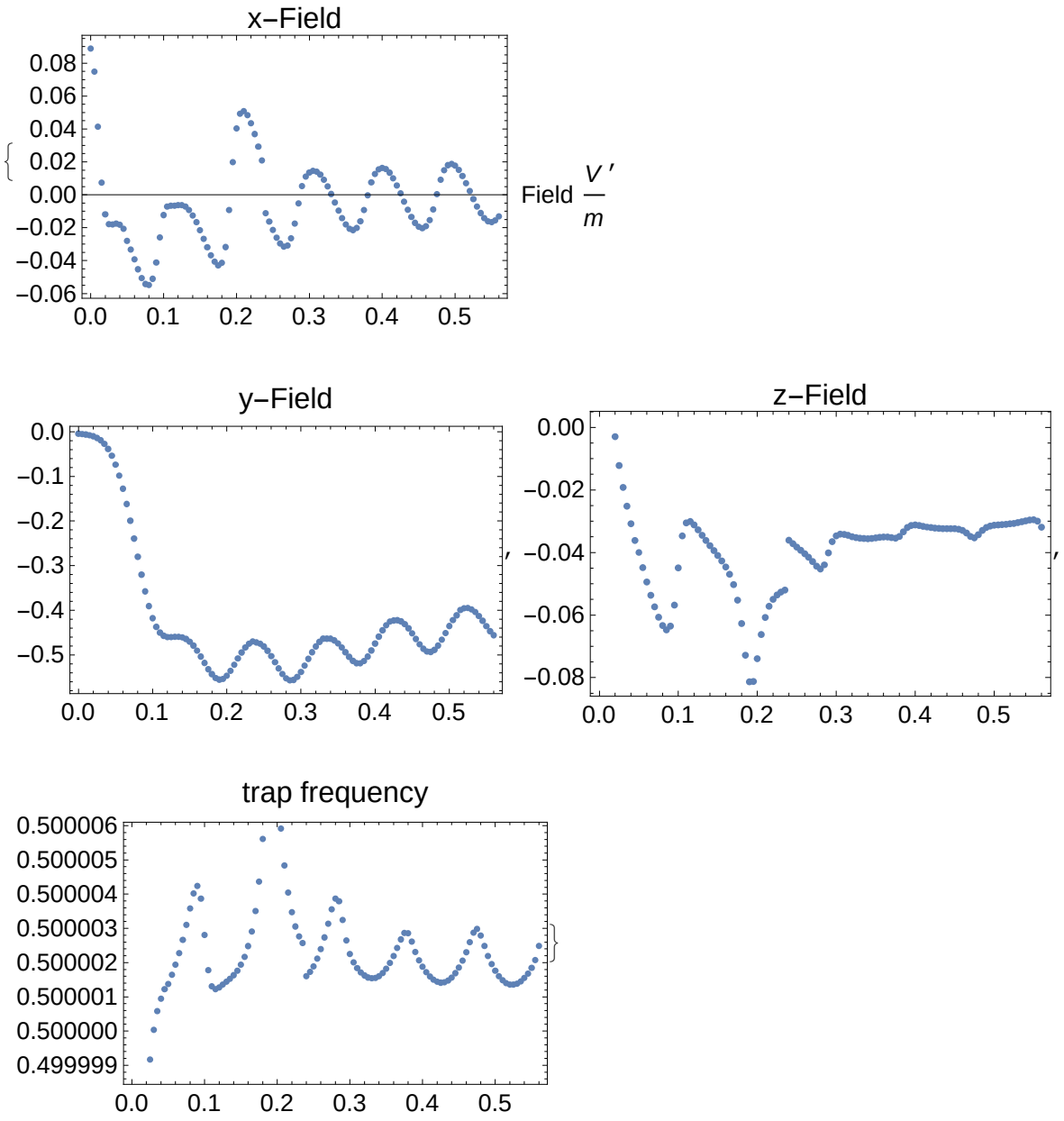
```
shuttleL1.png
```

Verification

```
With[{region = 2},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```



```
verificationPlots[solutionL1Rot, eMaskL1Rot, 2]
```

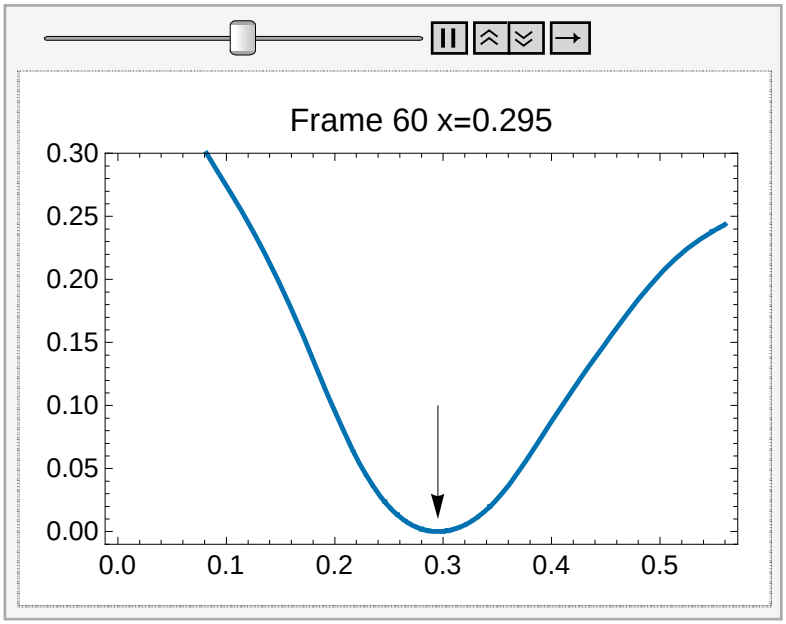


Solution for loading leg L2 (bottom left) with rotation

```
targetvoltageL2Rot[pos_] := targetvoltageL2[pos] + rotationVoltages
eMaskL2Rot[pos_] := eMaskL2[pos] + rotationVoltages
```

```
targetvoltageL2[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 6],
     ConstantArray[0, 10], 1, 1, ConstantArray[0, 12]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];

outerVoltage = -2;
outerVoltageL = -3;

eMaskL2[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 6],
     Array[s, 2], ConstantArray[0, 2], Array[s, 8, 3], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 4], Array[s, 2], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 8, 3], ConstantArray[0, 8]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 2], Array[s, 4], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 6, 5], ConstantArray[0, 10]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     Array[s, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 8],
     Array[s, 4, 7], ConstantArray[0, 12]}], xpos < 0.650, Flatten[
    {ConstantArray[0, 16], outerVoltageL, outerVoltageL, Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];
```

```
vMaskL2[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL2Rot = Table[(globalProgress = (posx - xmin[3]) / (xmax[3] - xmin[3]);
    Flatten[{posx,
      NMinimize[WeightFunction[eMaskL2Rot[posx], posx, 0.5, targetvoltageL2Rot, 3],
       vMaskL2[posx]]}, 1]), {posx, xmin[3], xmax[3], 0.005}];

Length[solutionL2]
113

ProgressIndicator[Dynamic[globalProgress]]

writeSolution["ShuttleL2Rot.txt", solutionL2Rot, eMaskL2Rot, 1]

allFramesL2Rot = solutionFrames[solutionL2Rot, eMaskL2Rot, 3];

ListAnimate[allFramesL2Rot, 5]
```



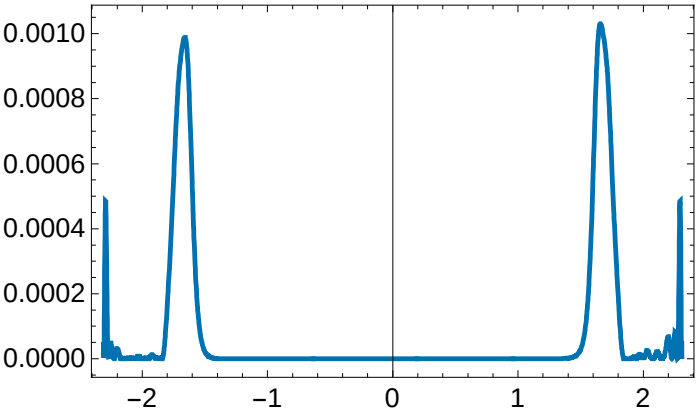

```
Export["shuttleL2Rot.swf", allFramesL2Rot, "FrameRate" → 5]
shuttleL2.swf

Export["shuttleL2Rot.cdf", ListAnimate[allFramesL2Rot, 5]]
shuttleL2.cdf
```

```
Export["shuttleL2Rot.png", allFramesL2Rot[[1]]]
```

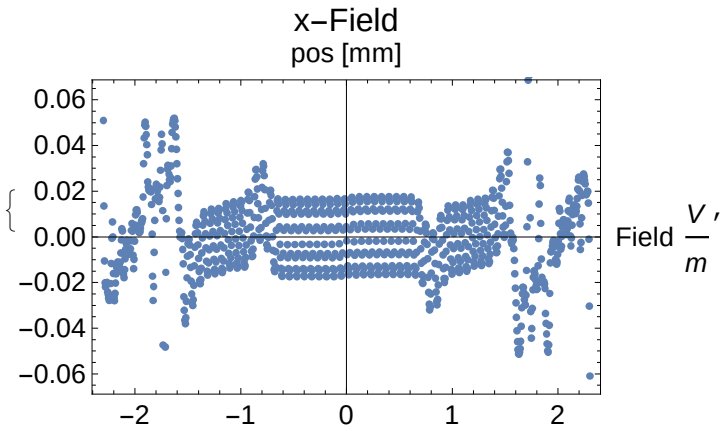
```
shuttleL2.png
```

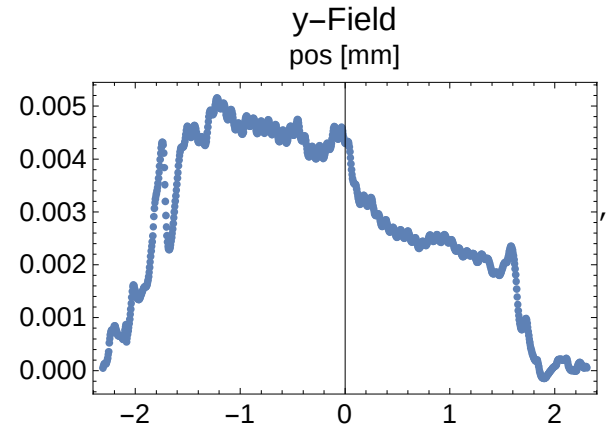
Verification

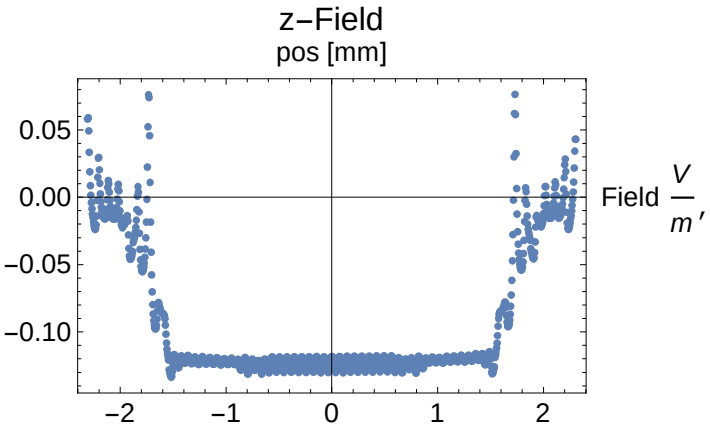
```
With[{region = 3},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```

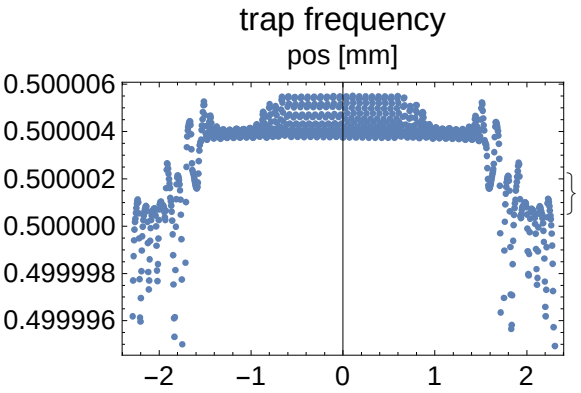



```
verificationPlots[solutionL2Rot, eMaskL2Rot, 3]
```









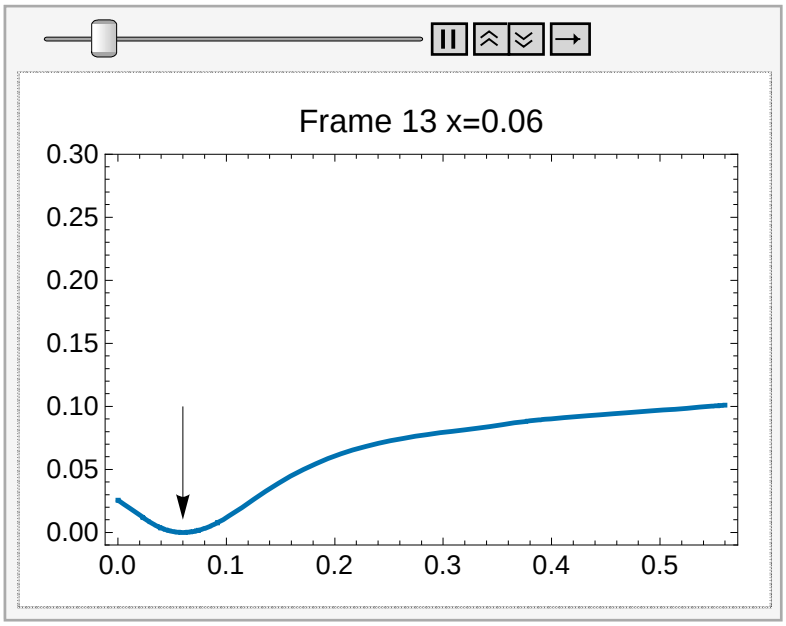
Solution for loading leg L3 (bottom right) with rotation

```
targetvoltageL3Rot[pos_] := targetvoltageL3[pos] + rotationVoltages
eMaskL3Rot[pos_] := eMaskL3[pos] + rotationVoltages

targetvoltageL3[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 6],
     ConstantArray[0, 10], 1, 1, ConstantArray[0, 12]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 8], 1, 1, ConstantArray[0, 14]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];
```

```
eMaskL3[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 6],
     Array[s, 2], ConstantArray[0, 2], Array[s, 8, 3], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14],
     outerVoltageL, outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 6], Array[s, 12], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 4], Array[s, 2], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 8, 3], ConstantArray[0, 8]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     ConstantArray[0, 2], Array[s, 4], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     ConstantArray[0, 8], Array[s, 6, 5], ConstantArray[0, 10]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 16], outerVoltageL, outerVoltageL,
     Array[s, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 8],
     Array[s, 4, 7], ConstantArray[0, 12]}], xpos < 0.650, Flatten[
    {ConstantArray[0, 16], outerVoltageL, outerVoltageL, Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL3[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL3Rot =
  Table[(globalProgress = (posx - xmin[4]) / (xmax[4] - xmin[4]); Flatten[{posx,
      NMinimize[WeightFunction[eMaskL3Rot[posx], posx, 0.5, targetvoltageL3Rot, 4],
       vMaskL3[posx]]}, 1]), {posx, xmin[4], xmax[4], 0.005}];

Length[solutionL3]
113

writeSolution["ShuttleL3Rot.txt", solutionL3Rot, eMaskL3Rot, 1]

allFramesL3Rot = solutionFrames[solutionL3Rot, eMaskL3Rot, 4];
$Aborted
```

```
ProgressIndicator[Dynamic[globalProgress]]
```

```
ListAnimate[allFramesL3Rot, 5]
```




```
Export["shuttleL3Rot.swf", allFramesL3Rot, "FrameRate" → 5]
```

shuttleL3.swf

```
Export["shuttleL3Rot.cdf", ListAnimate[allFramesL3Rot, 5]]
```

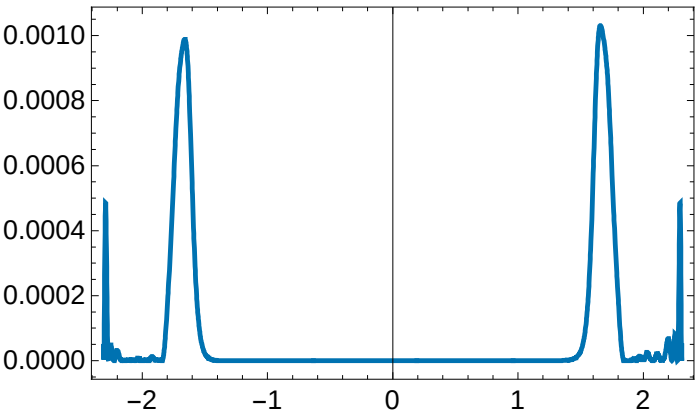
shuttleL3.cdf

```
Export["shuttleL3Rot.png", allFramesL3Rot[[1]]]
```

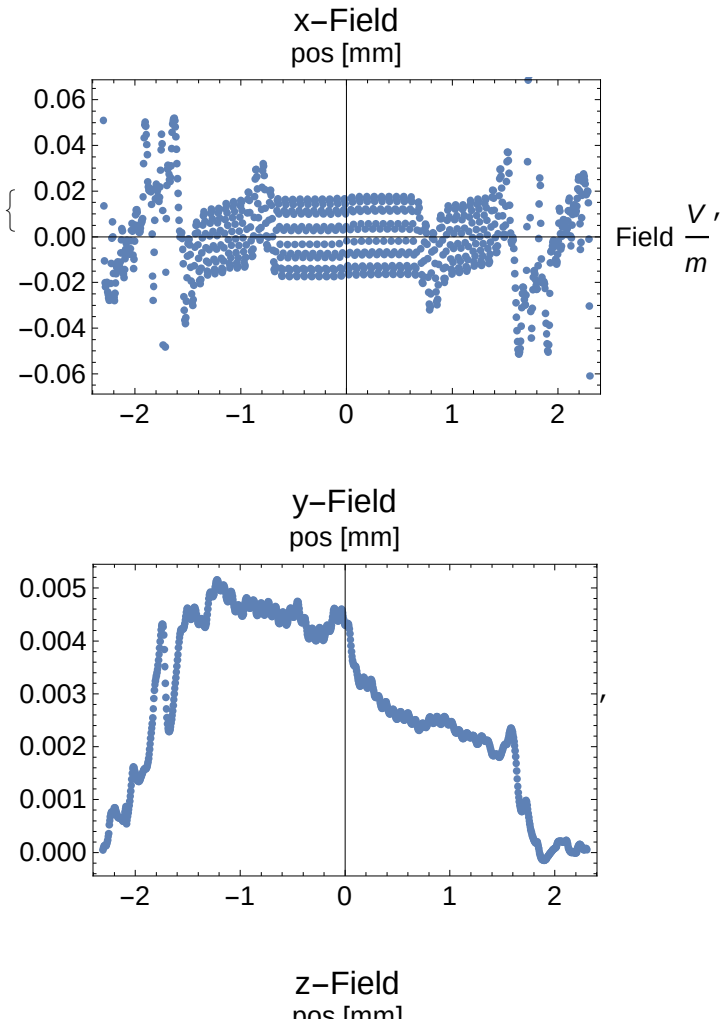
shuttleL3.png

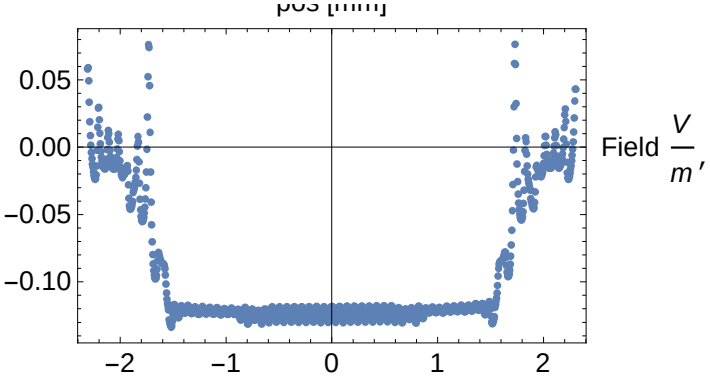
Verification

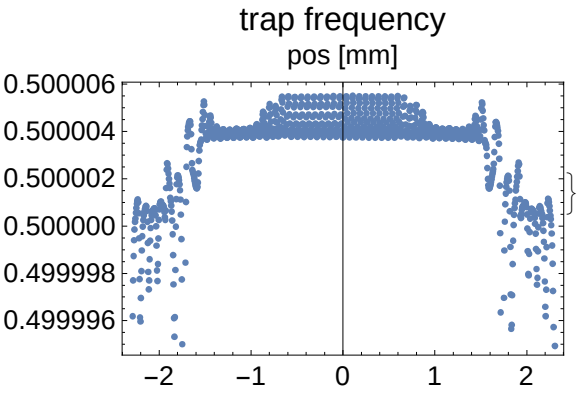
```
With[{region = 4},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```




```
verificationPlots[solutionL3Rot, eMaskL3Rot, 4]
```







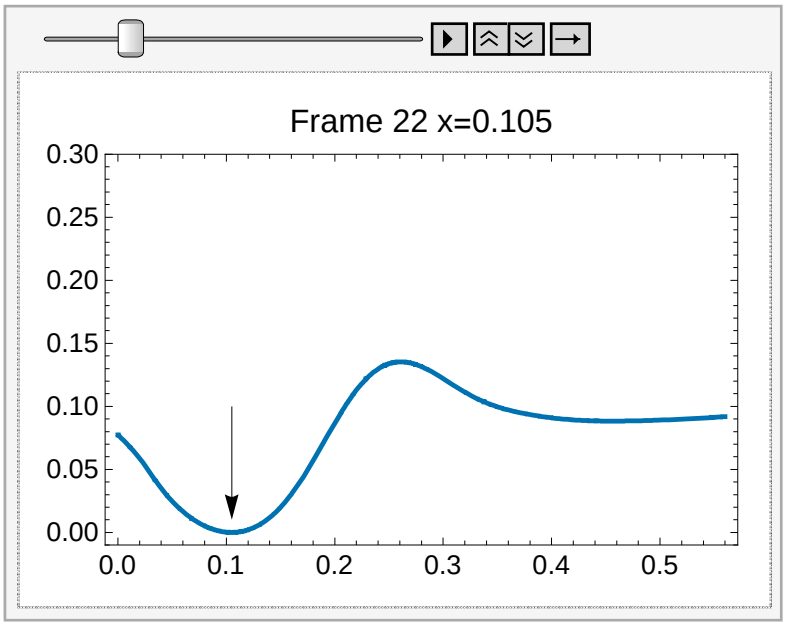
Solution for loading leg L4 (top right) with rotation

```
targetvoltageL4Rot[pos_] := targetvoltageL4[pos] + rotationVoltages
eMaskL4Rot[pos_] := eMaskL4[pos] + rotationVoltages

targetvoltageL4[pos_] =
  Which[pos < 0.0494, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], ConstantArray[0, 2], 1, 1, ConstantArray[0, 20]}],
   pos < 0.1446, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.2389, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1,
     ConstantArray[0, 6], 1, 1, ConstantArray[0, 22]}],
   pos < 0.650, Flatten[{ConstantArray[0, 26], ConstantArray[0, 12],
     ConstantArray[0, 14], ConstantArray[0, 12], -1, -1, ConstantArray[0, 30]}]];
```

```
eMaskL4[xpos_] = Which[xpos < 0.0494,
   Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL, outerVoltageL,
     ConstantArray[0, 6], outerVoltageL, outerVoltageL, ConstantArray[0, 38],
     outerVoltage, outerVoltage, ConstantArray[0, 6], ConstantArray[0, 2],
     Array[s, 6], ConstantArray[0, 6], Array[s, 4, 7], ConstantArray[0, 6]}],
   xpos < 0.1446, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.2389, Flatten[{ConstantArray[0, 2], ConstantArray[0, 14], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8], ConstantArray[0, 6], Array[s, 4, 9], ConstantArray[0, 6]}],
   xpos < 0.3333, Flatten[{ConstantArray[0, 14], Array[s, 2], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 8, 3], ConstantArray[0, 10], ConstantArray[0, 6]}],
   xpos < 0.4233, Flatten[{ConstantArray[0, 12], Array[s, 4], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 6, 5], ConstantArray[0, 12], ConstantArray[0, 6]}],
   xpos < 0.5133, Flatten[{ConstantArray[0, 10], Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL, outerVoltageL,
     ConstantArray[0, 38], outerVoltage, outerVoltage, ConstantArray[0, 6],
     Array[s, 4, 7], ConstantArray[0, 10], ConstantArray[0, 10]}],
   xpos < 0.650, Flatten[{ConstantArray[0, 10], Array[s, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 6], outerVoltageL,
     outerVoltageL, ConstantArray[0, 38], outerVoltage, outerVoltage,
     ConstantArray[0, 6], ConstantArray[0, 18], ConstantArray[0, 6]}]];

vMaskL4[xpos_] = Which[xpos < 0.0494, Array[s, 10],
   xpos < 0.1446, Array[s, 12],
   xpos < 0.2389, Array[s, 12],
   xpos < 0.3333, Array[s, 10],
   xpos < 0.4233, Array[s, 10],
   xpos < 0.5133, Array[s, 10],
   xpos < 0.650, Array[s, 6]];

solutionL4Rot =
  Table[(globalProgress = (posx - xmin[5]) / (xmax[5] - xmin[5]); Flatten[{posx,
      NMinimize[WeightFunction[eMaskL4Rot[posx], posx, 0.5, targetvoltageL4Rot, 5],
       vMaskL4[posx]]}, 1]), {posx, xmin[5], xmax[5], 0.005}];

Length[solutionL4Rot]

113

writeSolution["ShuttleL4Rot.txt", solutionL4Rot, eMaskL4Rot, 1]

allFramesL4Rot = solutionFrames[solutionL4Rot, eMaskL4Rot, 5];
```

```
ProgressIndicator[Dynamic[globalProgress]]
```

```
ListAnimate[allFramesL4Rot, 5]
```




```
Export["shuttleL4Rot.swf", allFramesL4Rot, "FrameRate" → 5]
```

shuttleL4.swf

```
Export["shuttleL4Rot.cdf", ListAnimate[allFramesL4Rot, 5]]
```

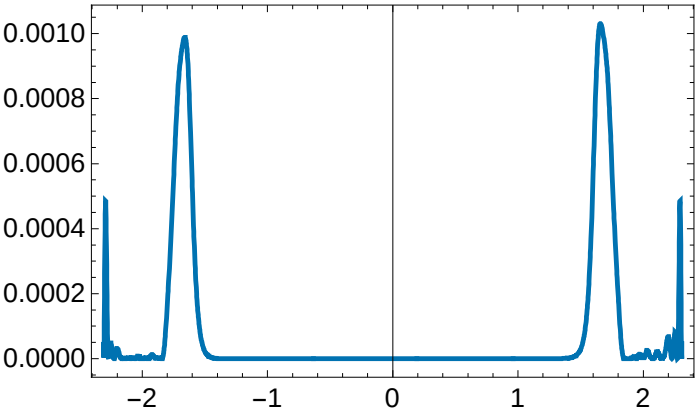
shuttleL4.cdf

```
Export["shuttleL4Rot.png", allFramesL4Rot[[1]]]
```

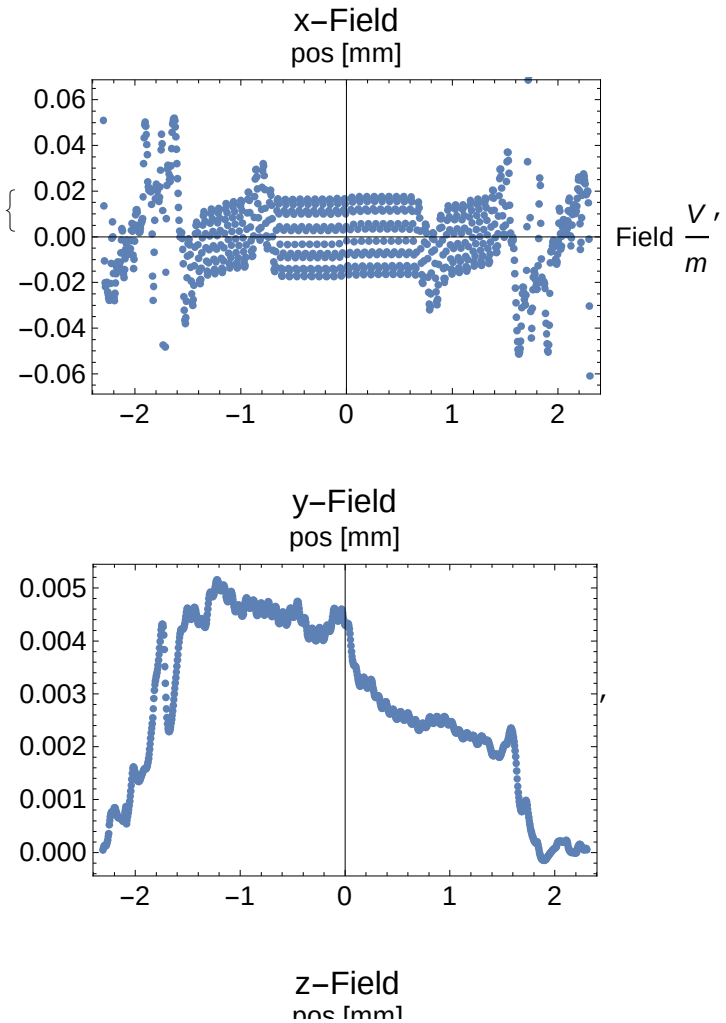
shuttleL4.png

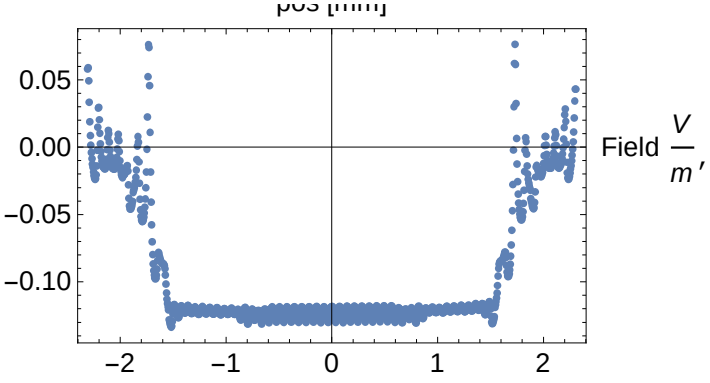
Verification

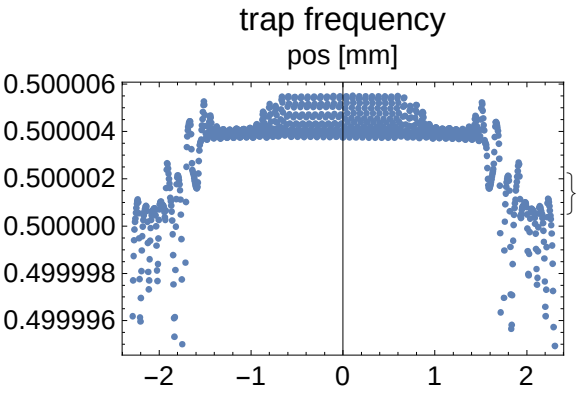
```
With[{region = 5},
 Plot[Evaluate[potentialrf[posx, nodeY[posx, region], nodeZ[posx, region], region]],
  {posx, xmin[region], xmax[region]}, PlotRange → All]]
```




```
verificationPlots[solutionL4Rot, eMaskL4Rot, 5]
```




0.05
0.00
−0.05
−0.10
−2
−1
0
1
2
Field $\frac{V}{m}$,


trap frequency
pos [mm]
0.500006
0.500004
0.500002
0.500000
0.499998
0.499996
−2
−1
0
1
2